2003-08-20

A Study of Radiofrequency Cardiac Ablation Using Analytical and Numerical Techniques

Ryan Todd Roper
Brigham Young University - Provo

A STUDY OF RADIOFREQUENCY CARDIAC ABLATION USING ANALYTICAL AND NUMERICAL TECHNIQUES

by

Ryan Todd Roper

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering

Brigham Young University

December 2003

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Ryan Todd Roper

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Matthew R. Jones, Chair
Date	R. Daniel Maynes
Date	Brent W. Webb

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Ryan Todd Roper in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

_____ _____

Date

Matthew R. Jones
Chair, Graduate Committee

Accepted for the Department

_____

Brent L. Adams
Graduate Coordinator

Accepted for the College

_____

Douglas M. Chabries
Dean, College of Engineering and Technology

ABSTRACT

A STUDY OF RADIOFREQUENCY CARDIAC ABLATION USING ANALYTICAL AND NUMERICAL TECHNIQUES

Ryan Todd Roper

Department of Mechanical Engineering

Master of Science

Studies on radiofrequency (RF) ablation are often aimed at accurately predicting tissue temperature distributions by numerical solution of the bioheat equation. This thesis describes the development of an analytical solution to serve as a benchmark for subsequent numerical solutions. The solution, which was obtained using integral transforms, has the form of a surface integral nested within another surface integral. An integration routine capable of evaluating such integrals was developed and a C program was written to implement this routine. The surface integration routine was validated using a surface integral with a known analytical solution. The routine was, then, used to generate temperature profiles at various times and for different convection coefficients. To further validate the numerical methods used to obtain temperature profiles, a numerical model was developed with the same approximations used in obtaining the analytical solution. Results of the analytical and numerical solutions match very closely.

In addition, three numerical models were developed to assess the validity of some of the assumptions used in obtaining the analytical solution. For each numerical model, one or two of the assumptions used in the analytical model were relaxed to better assess the degree to which they influence results. The results indicate that (1) conduction of heat into the electrode significantly affects lesion size, (2) temperature distributions can be assumed to be axisymmetric, and (3) lesion size and maximum temperature are strongly influenced by the temperature-dependence of electrical conductivity. These conclusions are consistent with results from previous studies on radiofrequency cardiac ablation.

ACKNOWLEDGEMENTS

I would, first, like to thank my parents for teaching me to take education seriously and for always expressing their confidence in my ability to succeed. I greatly appreciate their interest in my educational experiences and accomplishments and would like to acknowledge their hand in my success.

I am grateful for the valuable experiences I have had while working on this research project. I recognize that the value of these experiences is, in large part, due to the guidance of my advisor, Dr. Matthew Jones. I believe the counsel and direction he has given me during this time has better prepared me for future academic and professional endeavors.

I would like to thank my wife, Laura, for enduring so much as I have completed my research and the writing of this thesis. This is as much her accomplishment as it is mine. Not only has she felt all my stress, but she has, also, taken care of our daughter, Katelyn, with less help from me than should be expected. I thank both Laura and Katelyn for reminding me what is most important in life.

Finally, I feel it especially important to acknowledge the help of my Heavenly Father. Without some particularly valuable insights and ideas from Him, I do not believe this research would have been completed in the time that it has been.

TABLE OF CONTENTS

ABSTRACT iv

ACKNOWLEDGEMENTS vi

LIST OF TABLES ix

LIST OF FIGURES xi

NOMENCLATURE xvi

CHAPTER 1 – INTRODUCTION 1

1.1 Background 1

1.2 Objectives 4

CHAPTER 2 – ANALYTICAL SOLUTION OF THE BIOHEAT EQUATION 7

2.1 Formulation of the Ablation Model 7

2.2 Integral Transforms Method 12

2.3 Analytical Solution for the Voltage Field 14

2.4 Analytical Solution to the Bioheat Equation 18

CHAPTER 3 – EVALUATION OF THE ANALYTICAL SOLUTION 25

3.1 Early Attempts 25

3.2 Development of an Alternate Approach 31

3.3 Implementation of the Surface Integration Routine 36

3.4 Flow Chart for the Program Used to Generate Temperature Profiles 41

3.5 Reduction of Computation Time 47

CHAPTER 4 – TEMPERATURE PROFILES 51

4.1 Analytical Results 51

4.2 Numerical Results 58

CHAPTER 5 – NUMERICAL VALIDATION OF MODEL ASSUMPTIONS 65

5.1 Discussion of Analytical Model Assumptions 65

5.2 Numerical Model Descriptions 66

5.3 Determination of Appropriate Velocity Profiles 69

5.4 Model 1 – Actual Flow with Electrode Omitted 75

5.5 Model 2 – Actual Flow with Electrode Included 81

5.6 Model 3 – Temperature-dependent Electrical Conductivity 88

CHAPTER 6 – CONCLUSIONS AND RECOMMENDATIONS 99

REFERENCES 105

APPENDIX A – DERIVATION OF THE APPROPRIATE FORM FOR THE FOURIER TRANSFORM KERNEL 109

APPENDIX B – C CODE USED IN EVALUATING THE ANALYTICAL SOLUTION TO THE BIOHEAT EQUATION 113

APPENDIX C – USER-DEFINED FUNCTIONS USED FOR NUMERICAL MODELING 131

LIST OF TABLES

Table 3.1 Validation of numerical integration by comparison with an exact solution. 39

Table 4.1 Cardiac Tissue Properties [7-8]. 51

Table 5.1 Steady-state heat transfer rates through a 1 cm by 0.5 cm surface section at the ablation site, calculated from the analytical solution. This was done for three convection coefficients: 1000, 2000, and 4000 W/m^{2}K. 71

Table 5.2 Flow conditions corresponding to convective heat transfer coefficients of 1000, 2000, and 4000 W/m^{2}K..... 73

Table 5.3 Comparison of heat transfer rates obtained analytically and numerically for convection coefficients of 1000, 2000, and 4000 W/m^{2}K. 74

Table 5.4 Results of a grid-refinement study that compares heat transfer rates near the ablation electrode. The boundary layer thickness, δ, and free stream velocity, u_∞, of the blood flow are 1.6 cm and 0.2 m/s, respectively. 74

Table 5.5 Time step sizes used in the temporal grid refinement study for Model 1. Heat transfer rates through the small surface section were compared using flow conditions for a convection coefficient of 1000 W/m^{2}K. 76

Table 5.6 Heat transfer rates through the small surface section for different step sizes. Flow conditions were used that correspond to a convection coefficient of 1000 W/m^{2}K. 77

Table 5.7 Ablation electrode properties (Pt-Ir) [5] 82

Table 5.8 Results of a grid-refinement study that compares maximum tissue temperature. An ablation time of 60 seconds (Δt = 2 sec) and flow conditions corresponding to a convection coefficient of 2000 W/m^{2}K were used..... 83

LIST OF FIGURES

Figure 1.1 An illustration showing the process by which an electrode catheter is inserted through a blood vessel into the heart to resistively heat and destroy the affected cardiac tissue [1]. 1

Figure 2.1 Schematic depicting details of the ablation model. Included are the thermal boundary conditions. 8

Figure 2.2 Schematic showing the boundary conditions used in solving for the electric potential distribution. 11

Figure 2.3 Contour plot showing the dimensionless heat generation near the electrode surface. 18

Figure 3.1 Schematic of numerical integration using the trapezoidal method. 28

Figure 3.2 Schematic illustrating how a surface integral may be numerically evaluated. 33

Figure 3.3 Schematic illustrating the process by which surface integrals are evaluated. Row volumes are obtained by summing section volumes along the row while the total volume is obtained by summing the row volumes. 37

Figure 3.4 Results of an optimization study to determine an appropriate value for QMAX. Equation (2.70) (the Hankel and Fourier transforms of $Q(\rho,\zeta)$) was evaluated using different values of QMAX for several different values of γ and β. Results were normalized using the values obtained for QMAX = 20. 40

Figure 3.5 Surface plots of the dimensionless heat generation in the immediate vicinity of the truncated singularity (QMAX = 2000). The range over which the heat generation is plotted in (a) is four times that of (b). 41

Figure 3.6 A flow chart outlining the sequence of function calls required to generate a dimensionless temperature profile. 42

Figure 4.1 Dimensionless temperature profiles for h = 1000 W/m^{2}K and (a) τ = 0.3, (b) τ = 2.0, and (c) τ = 5.5. 53

Figure 4.2 Dimensionless temperature profiles for h = 2000 W/m^{2}K and (a) τ = 0.3, (b) τ = 2.0, and (c) τ = 5.5. 54

Figure 4.3 Dimensionless temperature profiles for h = 4000 W/m^{2}K and (a) τ = 0.3, (b) τ = 2.0, and (c) τ = 5.5. 55

Figure 4.4 An image showing the general features of the mesh used for the numerical model. The tissue section has two symmetry surfaces (sides), an adiabatic surface (bottom), and a surface with a convective boundary condition (top). The lengths of the straight edges are all 2 cm. 59

Figure 4.5 Dimensionless temperature profiles for τ = 5.5 and h = 2000 W/m^{2}K (a) along the surface and (b) along the axis. 60

Figure 4.6 Dimensionless surface temperature profile for τ = 5.5 and h = 2000 W/m^{2}K near the electrode edge. 60

Figure 4.7 Dimensionless temperature profiles for τ = 5.5 and h = 2000 W/m^{2}K (a) predicted numerically and (b) predicted analytically. 61

Figure 4.8 Plot of surface temperatures showing that the numerical results approach those of the analytical solution as the mesh is refined near the electrode. 62

Figure 5.1 Normalized velocity profiles corresponding to three different convection coefficients. The free stream velocities (u_∞) for the profiles corresponding to 1000, 2000, and 4000 W/m^{2}K, are 0.21, 0.40, and 1.30 m/s, respectively. 73

Figure 5.2 Schematic showing the main features of numerical model 1 with dimensions. The boundary layer thickness and free stream velocity of the blood are represented by δ and u_∞, respectively. 75

Figure 5.3 Temperature profiles obtained numerically for the case where δ = 2 cm and u_∞ = 0.21 m/s for ablation times of (a) 3.2 sec, (b) 22 sec, and (c) 60 sec. Also shown, in gray contours, are temperature profiles obtained analytically for the corresponding convection coefficient of 1000 W/m^{2}K..... 78

Figure 5.4 Temperature profiles obtained numerically for the case where δ = 0.40 cm and u_∞ = 0.40 m/s for ablation times of (a) 3.2 sec, (b) 22 sec, (c) and 60 sec. Also shown, in gray contours, are temperature profiles obtained analytically for the corresponding convection coefficient of 2000 W/m^{2}K..... 79

Figure 5.5 Temperature profiles obtained numerically for the case where δ = 0.10 cm and u_∞ = 1.30 m/s for ablation times of (a) 3.2 sec, (b) 22 sec, and (c) 60 sec. Also shown, in gray contours, are temperature profiles obtained analytically for the corresponding convection coefficient of 4000 W/m^{2}K..... 80

Figure 5.6 Schematic showing the main features of numerical model 2 with dimensions. The boundary layer thickness and free stream velocity of the blood are represented by δ and u_∞, respectively. 82

Figure 5.7 Temperature profiles for an ablation time of 60 seconds (Δt = 2 sec) using flow conditions that correspond to a convection coefficient of 2000 W/m^{2}K. The two profiles shown were obtained using grids of 150 124 and 336 681 cells. 84

Figure 5.8 Results obtained for Model 2 for an ablation time of 60 seconds and convection coefficients of (a) 1000, (b) 2000, and (c) 4000 W/m^{2}K. Also included are the corresponding analytical results shown as gray contours..... 85

Figure 5.9 Results obtained for Model 2 for an ablation time of 60 seconds and convection coefficients of (a) 1000, (b) 2000, and (c) 4000 W/m^{2}K. Also included are the corresponding Model 1 results shown as gray contours..... 86

Figure 5.10 Schematic showing the main features of numerical model 3, including (a) thermal and (b) electrical boundary conditions. 88

Figure 5.11 Heat generation rates predicted analytically and numerically for the case of a temperature-independent electrical conductivity. 91

Figure 5.12 Temperature contours generated for a convection coefficient of 2000 W/m^{2}K and an ablation time of 60 seconds. The dashed contours were generated from a numerical model in which the heat generation rate was calculated numerically. The solid lines are contours obtained analytically. 91

Figure 5.13 Contour plots for a convection coefficient of 1000 W/m^{2}K and an ablation time of 60 seconds assuming (a) a constant electrical conductivity and (b) an electrical conductivity that increases 2%/°C. 92

Figure 5.14 Contour plots for a convection coefficient of 2000 W/m^{2}K and an ablation time of 60 seconds assuming (a) a constant electrical conductivity and (b) an electrical conductivity that increases 2%/°C. 93

Figure 5.15 Contour plots for a convection coefficient of 4000 W/m^{2}K and an ablation time of 60 seconds assuming (a) a constant electrical conductivity and (b) an electrical conductivity that increases 2%/°C. 94

Figure 5.16 Electric potential distribution in the tissue along the axis (ρ = 0) after 60 seconds of ablation using a convection coefficient of 2000 W/m^{2}K. The dashed line shows the distribution assuming a temperature dependence of +2%/°C for the electrical conductivity and the solid line for a constant conductivity..... 96

Figure 5.17 Heat generation rate in the tissue along the axis (ρ = 0) after 60 seconds of ablation using a convection coefficient of 2000 W/m^{2}K. The dashed line shows the distribution assuming a temperature dependence of +2%/°C for the electrical conductivity and the solid line for a constant conductivity..... 97

NOMENCLATURE

Bi	Biot number, $Bi = hR/k$
c	Specific heat
h	Convective heat transfer coefficient
J_o	Zero-order Bessel function of the first kind
k	Conductive heat transfer coefficient
K	Kernel for the Fourier transform
$\dot{q}$	Energy generation
q''	Tissue surface heat flux
Q	Dimensionless energy generation, $Q = R^2\dot{q}/kT_0$
Q''	Dimensionless tissue surface heat flux, $Q'' = Rq''/kT_0$
r	Radial coordinate
R	Radius of ablation electrode
t	Time
T	Temperature
T_o	Body temperature
v	Root mean squared voltage
v_o	Root mean squared voltage at the electrode surface
V	Dimensionless root mean squared voltage, $V = v/v_0$
z	Axial coordinate

α Thermal diffusivity

β Transform variable for the Fourier transform

γ Transform variable for the Hankel transform

θ Dimensionless temperature, $\theta = (T - T_0)/T_0$

ρ Dimensionless radial coordinate, $\rho = r/R$

ρ_t Tissue density

σ Electrical conductivity

τ Dimensionless time, $\tau = t\alpha/R^2$

ζ Dimensionless axial coordinate, $\zeta = z/R$

CHAPTER 1 – INTRODUCTION

1.1 Background

Radiofrequency (RF) ablation is a minimally-invasive surgical procedure that has been very successful in treating certain types of cardiac arrhythmia. This procedure consists of inserting an electrode catheter through a blood vessel into the heart of a patient where radiofrequency current is applied to the site responsible for initiation of the arrhythmic contractions. A small lesion is produced as a result of resistive heating, thus eliminating the source of the arrhythmia. This process is illustrated in Fig. 1.1.

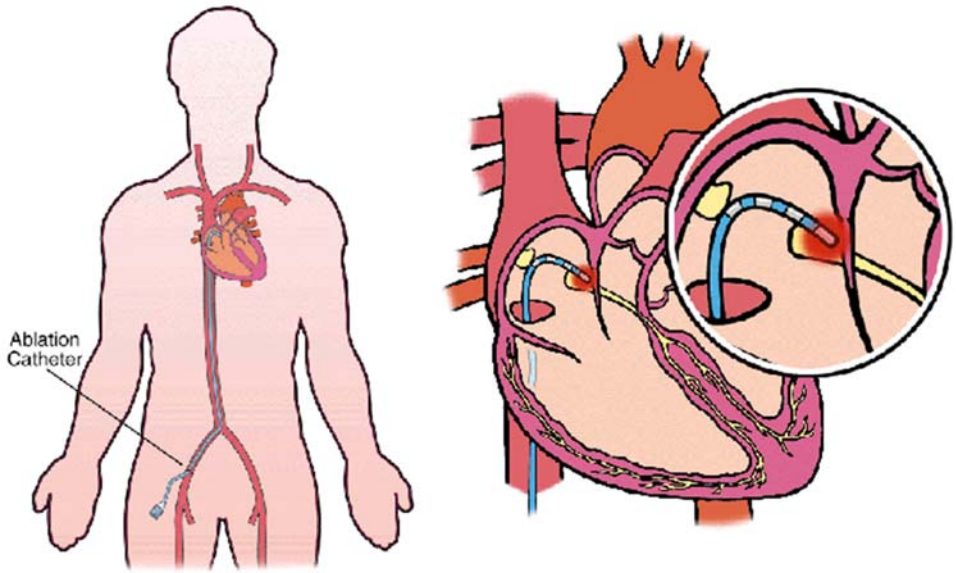

Figure 1.1 An illustration showing the process by which an electrode catheter is inserted through a blood vessel into the heart to resistively heat and destroy the affected cardiac tissue [1].

While this procedure has been highly effective in treating some types of arrhythmia, many studies are aimed at gaining a better understanding of the factors

involved in lesion formation. The ultimate objective of this research is to further improve success rates and to make the procedure applicable to a wider range of arrhythmias. Two problems in particular may occur when using this method of treatment. First, there is the potential of damaging non-target tissue that is critical for normal function of the heart. On the other hand, it is thought that target tissue on the border of the lesion that is not successfully ablated can become permanently dysfunctional [2]. Either of these outcomes may necessitate the use of a cardiac assist device to help the patient's heart to work properly. Research on RF ablation is motivated by the idea that such complications may be avoided by developing accurate models that predict lesion geometry based on such factors as applied power, ablation time, and location of the ablation site within the heart.

The overriding need in predicting lesion geometry is a better understanding of the heat transfer processes that occur during RF ablation. Detailed modeling of conductive heat transfer within the tissue and convective heat transfer from the surface to the blood pool has already been performed [3-7]. One study shows that, while accurate prediction of both lesion size and blood temperature require detailed thermal and fluid models, the lesion geometry alone can be accurately predicted using a simple thermal model based on a convective boundary condition at the tissue surface, where the convection coefficient has been experimentally determined [4]. This eliminates the need to solve the momentum transport equations for blood flow over the tissue surface and the problem is greatly simplified.

However, the accuracy of such a model depends largely on the accuracy with which the convection coefficient can be determined. According to Bhavaraju [8], the

value of this coefficient varies significantly (from less than 100 W/m^2K to almost 4000 W/m^2K) depending on the location within the heart. This is a result of large variations in blood flow characteristics throughout the heart. Therefore, even if convection coefficients can be accurately determined through experimentation, the significant variation in conditions throughout the heart makes it difficult to formulate a standard ablation protocol that can be used in treating patients. In other words, it is difficult to determine such parameters as the power level and the length of time required to produce a desired lesion shape and size because these depend strongly on flow characteristics immediately around the ablation site.

A potential solution to this problem involves the use of inverse heat transfer techniques to discover if a relationship exists between a set of easily measured tissue temperatures and the convective heat transfer coefficient. If tissue properties are known, the convection coefficient is the only remaining parameter necessary for predicting lesion shape and size, as long as an appropriate mathematical model can be developed. If a relationship can, in fact, be found between a set of easily measured tissue temperatures and the convection coefficient, then the need to experimentally determine the value of the convection coefficient would be eliminated. Measurement of a few temperatures near the electrode would be sufficient for a computer to construct the entire temperature profile within the tissue at any given time during the ablation procedure. Through a feedback loop, in which the profile is continually updated based on the measured temperatures, the power could be automatically turned off when the lesion reaches the desired size. Assuming sufficiently efficient algorithms could be designed to interpret the temperatures

measured near the electrode, this approach would provide a very simple and effective way of producing lesions of a desired size and shape.

The first step required for the solution of any inverse problem is the solution of the forward problem. In heat transfer problems involving human tissue, this requires solving the bioheat equation. Such a solution allows tissue temperature distributions to be obtained for different specified conditions. This has been done previously in several studies using finite element modeling techniques [3-7]. None of these studies, however, present a method for solving the bioheat equation analytically.

1.2 Objectives

The following is a summary of the objectives of this research:

(1) To develop an analytical solution to the bioheat equation that will serve as a benchmark for numerical studies of RF ablation. It was also anticipated that such a solution would provide insight needed to effectively solve the inverse heat transfer problem.

(2) To develop a means of efficiently and accurately evaluating the solution. This is essential if the solution is going to be used in solving the inverse heat transfer problem, since it must be evaluated for several times and convection coefficients.

(3) To assess the validity of approximations made in obtaining the analytical solution by conducting numerical studies. This is important since the quality of results obtained from an inverse heat transfer study depends on the degree to which the model used to solve the forward problem

represents the actual physical processes. The objective of these numerical studies was also to gain additional insights into RF ablation.

It is important to note that, while this work forms the foundation for a study to be conducted using inverse heat transfer techniques, no attempt has been made to solve the inverse problem in this thesis.

CHAPTER 2 – ANALYTICAL SOLUTION OF THE BIOHEAT EQUATION

2.1 Formulation of the Ablation Model

In heat transfer problems involving human tissue, the bioheat equation, which is shown as Eq. (2.1), may be solved to obtain tissue temperature distributions. In this equation, Q_m and Q_p represent the metabolic heat generation and the heat loss due to blood perfusion, respectively.

$$\nabla \cdot k\nabla T + \dot{q} + Q_m - Q_p = \rho_t c \frac{\partial T}{\partial t} \tag{2.1}$$

To develop an analytical solution to the bioheat equation, a flat-tipped electrode and a convective boundary condition at the tissue surface are assumed. While a round-tipped electrode is used in actual ablation procedures, it was necessary to assume a flat-tipped electrode to make the problem tractable. In addition, it was necessary to assume a uniform convective boundary condition instead of a mixed boundary condition at the surface to make the problem tractable. Therefore, this model does not account for conduction into the ablation electrode. To emphasize this point, the schematic shown in Fig. 2.1 does not include the electrode. The conductive heat loss that would occur at the tissue-electrode interface is assumed to be similar to that predicted using a convective boundary condition that is uniform over the entire tissue surface.

To account for resistive heating in the tissue produced by the electrode, an energy generation function is developed in Section 2.3. The electrical conductivity is assumed to be independent of temperature so that the energy generation is a function of location only

(The validity of this assumption will be discussed in Chapter 5). It is represented in Fig. 2.1 by the notation, $\dot{q}(r,z)$. The spatial variables, *r* and *z*, represent the radial and axial positions, respectively, in cylindrical coordinates. In developing this function, the electrode tip is modeled as a flat electrified disk.

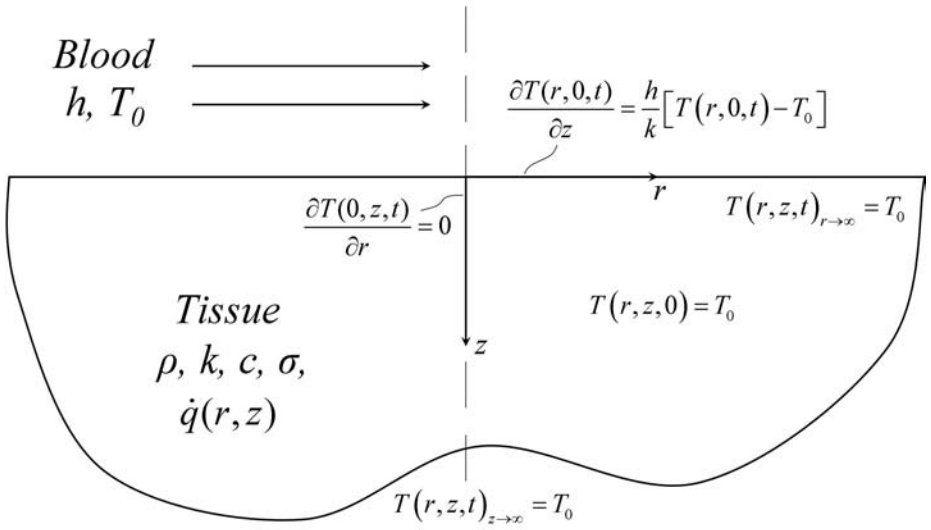

Figure 2.1 Schematic depicting details of the ablation model. Included are the thermal boundary conditions.

Typically, metabolic heat generation is neglected since it is small compared to the energy generated by resistive heating [5-7]. Likewise, heat loss due to blood perfusion is neglected since it is small relative to the heat lost at the tissue surface. With these approximations, the bioheat equation is identical to the heat equation:

$$\nabla \cdot k \nabla T + \dot{q} = \rho_t c \frac{\partial T}{\partial t} \tag{2.2}$$

In addition, if it is assumed that thermal conductivity is uniform and independent of temperature [7] and that the tissue temperature distribution is axisymmetric [4], the governing equation has the form shown in Eq. (2.3). Tungjitkusolmun et al. [7] show, by finite element analysis, that accounting for the temperature dependence of thermal

conductivity results in only a 1.5% change in lesion volume when the energy is applied in a power-controlled manner. If it is applied in a temperature-controlled manner, lesion volume only changes by 3.5%. The assumption of an axisymmetric temperature distribution has been studied by Jain and Wolf [4] by considering the effects of blood heating on lesion geometry. They show that, while some distortion of the lesion occurs, it is not significant. This will be further investigated in Chapter 5.

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right)+\frac{\partial^2 T}{\partial z^2}+\frac{\dot{q}}{k}=\frac{1}{\alpha}\frac{\partial T}{\partial t} \tag{2.3}$$

The initial condition and boundary conditions used to solve Eq. (2.3) are shown as Eqs. (2.4) – (2.6), below.

$$T\left(r,z,0\right)=T_0 \tag{2.4}$$

$$\left.\frac{\partial T}{\partial r}\right|_{r=0}=0 \qquad T\left(r,z,t\right)_{r\to\infty}=T_0 \tag{2.5}$$

$$\left.\frac{\partial T}{\partial z}\right|_{z=0}=\frac{h}{k}\left[T\left(r,0,t\right)-T_0\right] \qquad T\left(r,z,t\right)_{z\to\infty}=T_0 \tag{2.6}$$

Using dimensionless parameters as defined in the nomenclature, the governing equation, initial condition, and boundary conditions become:

$$\frac{1}{\rho}\frac{\partial}{\partial \rho}\left(\rho\frac{\partial \theta}{\partial \rho}\right)+\frac{\partial^2 \theta}{\partial \zeta^2}+Q=\frac{\partial \theta}{\partial \tau} \tag{2.7}$$

$$\theta(\rho,\zeta,0)=0 \tag{2.8}$$

$$\left.\frac{\partial\theta}{\partial\rho}\right|_{\rho=0} = 0$$
$$\theta(\rho,\zeta,\tau)_{\rho\to\infty} = 0 \tag{2.9}$$

$$\left.\frac{\partial\theta}{\partial\zeta}\right|_{\zeta=0} = Bi\theta(\rho,0,\tau)$$
$$\theta(\rho,\zeta,\tau)_{\zeta\to\infty} = 0 \tag{2.10}$$

Solution of Eq. (2.7) requires that an expression for the heat generation be developed. The heat generation is defined by the following equation [6]:

$$\dot{q}(r,z) = \sigma(\nabla v \cdot \nabla v) \tag{2.11}$$

Assuming an axisymmetric potential distribution, Eq. (2.11) may be rewritten as Eq. (2.12).

$$\dot{q}(r,z) = \sigma\left[\left(\frac{\partial v}{\partial r}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2\right] \tag{2.12}$$

Since the applied voltage oscillates at a high frequency (500 kHz to 1 MHz [4]), the voltage field can be modeled as steady and equal to the root mean squared (rms) voltage field [6]. An expression for the rms voltage field is found by solving the Laplace equation. This equation and its boundary conditions for the case of an electrified disk are shown in Eqs. (2.13) – (2.15) [9]. Figure 2.2 is a schematic showing the boundary conditions from Eqs. (2.14) and (2.15).

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v}{\partial r}\right) + \frac{\partial^2 v}{\partial z^2} = 0 \tag{2.13}$$

$$\left.\frac{\partial v}{\partial r}\right|_{r=0} = 0$$
$$v(r,z)_{r\to\infty} = 0 \tag{2.14}$$

$$v(r,0) = v_0 \qquad 0 \le r \le R$$
$$\left.\frac{\partial v}{\partial z}\right|_{z=0} = 0 \qquad r > R \tag{2.15}$$
$$v(r,z)_{z\to\infty} = 0$$

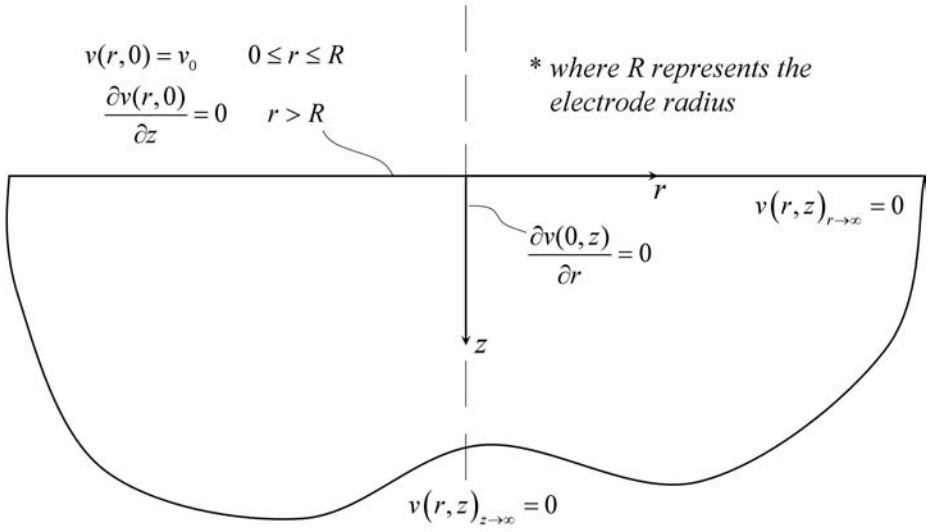

Figure 2.2 Schematic showing the boundary conditions used in solving for the electric potential distribution.

As was done with Eqs. (2.3) – (2.6), these equations may be written in dimensionless form using the dimensionless parameters defined in the nomenclature. These equations are shown as Eqs. (2.16) – (2.18).

$$\frac{1}{\rho}\frac{\partial}{\partial\rho}\left(\rho\frac{\partial V}{\partial\rho}\right) + \frac{\partial^2 V}{\partial\zeta^2} = 0 \tag{2.16}$$

$$\left.\frac{\partial V}{\partial\rho}\right|_{\rho=0} = 0$$
$$V(\rho,\zeta)_{\rho\to\infty} = 0 \tag{2.17}$$

$$
\begin{aligned}
V(\rho,0) &= 1 \qquad 0 \le \rho \le 1 \\
\left.\frac{\partial V}{\partial \zeta}\right|_{\zeta=0} &= 0 \qquad \rho > 1 \\
V(\rho,\zeta)_{\zeta\to\infty} &= 0
\end{aligned}
\tag{2.18}
$$

Finally, Eq. (2.12) may be non-dimensionalized to obtain an expression for the dimensionless heat generation, shown as Eq. (2.19).

$$
Q = \frac{R^2 \dot{q}}{kT_0} = \frac{\sigma v_0^2}{kT_0}\left[\left(\frac{\partial V}{\partial \rho}\right)^2 + \left(\frac{\partial V}{\partial \zeta}\right)^2\right] \tag{2.19}
$$

The following sections describe, in detail, methods for solving Eqs. (2.7) and (2.16) to obtain an expression for the dimensionless temperature as a function of location and time.

2.2 Integral Transforms Method

The integral transforms method of solving partial differential equations (PDE) consists of reducing a PDE to an ordinary differential equation (ODE) by removing the dependence on all but one of the independent variables. The ODE may, then, be solved using any appropriate solution method. Such a procedure may be used in solving Eqs. (2.7) and (2.16) to obtain expressions for the dimensionless temperature field and the dimensionless voltage field, respectively. In each case, a zero-order Hankel transform may be used to eliminate dependence on the dimensionless radial position. The zero-order Hankel transform and its inverse are defined in Eq. (2.20), where f is any arbitrary function of ρ and γ is the transform variable [9-10].

$$
\begin{aligned}
\overline{f}(\gamma) &\equiv \mathcal{H}_0\{f(\rho)\} \equiv \int_0^\infty \rho f(\rho) J_0(\gamma\rho)\, d\rho \\
f(\rho) &= \mathcal{H}_0^{-1}\{\overline{f}(\gamma)\} \equiv \int_0^\infty \gamma \overline{f}(\gamma) J_0(\gamma\rho)\, d\gamma
\end{aligned}
\tag{2.20}
$$

In the case of Eq. (2.16), a second-order, linear, homogeneous ODE is obtained that may be solved using any appropriate method. However, application of the Hankel transform to Eq. (2.7) results in another partial differential equation with independent variables ζ and τ. An ODE may be obtained by applying a Fourier transform to remove the dependence on ζ. The Fourier transform and its inverse are defined in Eq. (2.21), where g is any arbitrary function of ζ and β is the transform variable [9-10]. The function, $K(\beta,\zeta)$, is the kernel for the Fourier transform and its form is dependent upon the characteristics of the problem. For the case of a semi-infinite medium with a convective boundary condition at the surface, the kernel has the form shown in Eq. (2.22) [10] (see Appendix A for a derivation of this kernel).

$$\begin{aligned} \bar{g}(\beta) &\equiv \mathcal{F}\{g(\zeta)\} \equiv \int_0^\infty K(\beta,\zeta) g(\zeta) d\zeta \\ g(\zeta) &= \mathcal{F}^{-1}\{\bar{g}(\beta)\} \equiv \int_0^\infty K(\beta,\zeta) \bar{g}(\beta) d\beta \end{aligned} \tag{2.21}$$

$$K(\beta,\zeta) = \sqrt{\frac{2}{\pi}} \left[\frac{\beta \cos \beta\zeta + Bi \sin \beta\zeta}{\sqrt{\beta^2 + Bi^2}} \right] \tag{2.22}$$

Application of both the Hankel and Fourier Transforms to Eq. (2.7) results in a first-order, linear, non-homogeneous ODE that is readily solved. The final solution is obtained by application of the inverse Hankel and Fourier transforms.

Sections 2.3 and 2.4 show, in detail, the steps followed in obtaining analytical solutions to Eqs. (2.16) and (2.7), respectively, with the use of integral transforms. In addition, Section 2.3 shows the development of a dimensionless heat generation function from the solution to Eq. (2.16).

2.3 Analytical Solution for the Voltage Field

Applying the Hankel transform to every term in Eq. (2.16) results in the following equation:

$$\int_0^\infty \frac{\partial}{\partial\rho}\left(\rho\frac{\partial V}{\partial\rho}\right)J_0(\gamma\rho)\,d\rho+\int_0^\infty \rho\frac{\partial^2 V}{\partial\zeta^2}J_0(\gamma\rho)\,d\rho=0 \tag{2.23}$$

The second term is easily evaluated by reversing the order of operations and using the definition of the Hankel transform to obtain the result in Eq. (2.24).

$$\frac{\partial^2}{\partial\zeta^2}\int_0^\infty \rho V J_0(\gamma\rho)\,d\rho=\frac{\partial^2\overline{V}}{\partial\zeta^2} \tag{2.24}$$

The first term in Eq. (2.23) may be evaluated using integration by parts to obtain Eq. (2.25). Then, using the boundary conditions and the properties of Bessel functions shown in Eq. (2.26) as well as the definition of the Hankel transform in Eq. (2.20), the final result of integration by parts is Eq. (2.27).

$$\begin{aligned}\int_0^\infty \frac{\partial}{\partial\rho}\left(\rho\frac{\partial V}{\partial\rho}\right)J_0(\gamma\rho)\,d\rho=&\\ &\left[\rho\frac{\partial V}{\partial\rho}J_0(\gamma\rho)\right]_0^\infty+\gamma\left[\rho V J_1(\gamma\rho)\right]_0^\infty-\gamma^2\int_0^\infty \rho V J_0(\gamma\rho)\,d\rho\end{aligned} \tag{2.25}$$

$$\left.\frac{\partial V}{\partial\rho}\right|_{\rho=0}=\left.\frac{\partial V}{\partial\rho}\right|_{\rho\to\infty}=V(\rho,\zeta)_{\rho\to\infty}=J_0(\gamma\rho)_{\rho\to\infty}=J_1(0)=J_1(\gamma\rho)_{\rho\to\infty}=0 \tag{2.26}$$

$$\int_0^\infty \frac{\partial}{\partial\rho}\left(\rho\frac{\partial V}{\partial\rho}\right)J_0(\gamma\rho)\,d\rho=-\gamma^2\overline{V} \tag{2.27}$$

The ODE obtained by application of the Hankel transform is, therefore:

$$\frac{\partial^2\overline{V}}{\partial\zeta^2}-\gamma^2\overline{V}=0 \tag{2.28}$$

The solution to Eq. (2.28) has the form shown in Eq. (2.29). Recognizing that $\bar{V}$ must be bounded as ζ approaches infinity and that $B(\gamma)$ must, therefore, be zero, Eq. (2.30) is obtained by application of the inverse Hankel transform.

$$\bar{V} = A(\gamma)e^{-\gamma\zeta} + B(\gamma)e^{\gamma\zeta} \tag{2.29}$$

$$V(\rho,\zeta) = \int_0^\infty \gamma\left[A(\gamma)e^{-\gamma\zeta}\right]J_0(\gamma\rho)\,d\gamma \tag{2.30}$$

To obtain the final solution to Eq. (2.16), the coefficient, $A(\gamma)$, must be determined. Since the boundary conditions in ρ were used in evaluating the Hankel transform and the boundary condition at $\zeta \to \infty$ was used in determining $B(\gamma)$, the remaining boundary condition at $\zeta = 0$, shown in Eq. (2.18), is used in determining $A(\gamma)$. Because two separate conditions must be satisfied at $\zeta = 0$, the following two equations are obtained:

$$\int_0^\infty \gamma A(\gamma)J_0(\gamma\rho)\,d\gamma = 1 \qquad (0 \le \rho \le 1) \tag{2.31}$$

$$\int_0^\infty \gamma^2 A(\gamma)J_0(\gamma\rho)\,d\gamma = 0 \qquad (\rho > 1) \tag{2.32}$$

From a table of Hankel transforms [9], it is known that the Hankel transform of $(\sin\rho)/\rho$ for $\gamma > 1$ is zero and that the Hankel transform of $(\sin\rho)/\rho^2$ for $\gamma \le 1$ is equal to $\pi/2$. Recognizing that the form of the inverse Hankel transform is identical to the forward transform, it can be assumed that the inverse Hankel transform of $(\sin\gamma)/\gamma$ for $\rho > 1$ is zero and that the inverse Hankel transform of $(\sin\gamma)/\gamma^2$ for $\rho \le 1$ is $\pi/2$. Therefore, Eqs. (2.31) and (2.32) are satisfied if the following is true of $A(\gamma)$:

$$A(\gamma) = \frac{2}{\pi}\left[\frac{\sin\gamma}{\gamma^2}\right] \tag{2.33}$$

This may be seen in Eqs. (2.34) and (2.35).

$$\frac{2}{\pi}\int_0^\infty \gamma\left[\frac{\sin\gamma}{\gamma^2}\right]J_0(\gamma\rho)\,d\gamma=\frac{2}{\pi}\left(\frac{\pi}{2}\right)=1 \qquad (\rho\le 1) \tag{2.34}$$

$$\begin{aligned}&\frac{2}{\pi}\int_0^\infty \gamma^2\left[\frac{\sin\gamma}{\gamma^2}\right]J_0(\gamma\rho)\,d\gamma=\\ &\qquad\frac{2}{\pi}\int_0^\infty \gamma\left[\frac{\sin\gamma}{\gamma}\right]J_0(\gamma\rho)\,d\gamma=\frac{2}{\pi}(0)=0 \quad (\rho>1)\end{aligned} \tag{2.35}$$

Thus, the final solution to Eq. (2.16) is [9,11]:

$$V(\rho,\zeta)=\frac{2}{\pi}\int_0^\infty \frac{\sin\gamma}{\gamma}e^{-\gamma\zeta}J_0(\gamma\rho)\,d\gamma \tag{2.36}$$

The integral in Eq. (2.36) has a known analytical solution. Equation (2.37) shows this integral and its analytical solution in a general form [12].

$$\int_0^\infty \frac{\sin(ax)}{x}e^{-px}J_0(\lambda x)\,dx=\sin^{-1}\left(\frac{2a}{\sqrt{p^2+(a+\lambda)^2}+\sqrt{p^2+(a-\lambda)^2}}\right) \tag{2.37}$$

By inspection of Eqs. (2.36) and (2.37), it can be seen that $a=1$, $p=\zeta$, $x=\gamma$, and $\lambda=\rho$. Therefore, Eq. (2.38) shows an expression for the dimensionless electric potential distribution that is equivalent to Eq. (2.36) and is much more readily evaluated.

$$V(\rho,\zeta)=\frac{2}{\pi}\sin^{-1}\left(\frac{2}{\sqrt{\zeta^2+(1+\rho)^2}+\sqrt{\zeta^2+(1-\rho)^2}}\right) \tag{2.38}$$

With the above expression for the dimensionless rms electric potential distribution in the cardiac tissue, an expression for the dimensionless heat generation function may be developed using Eq. (2.19). Expressions for the derivatives of the dimensionless potential field with respect to ρ and ζ are shown as Eqs. (2.39) and (2.40), respectively. The dimensionless heat generation function may be obtained using Eqs. (2.19), (2.39),

and (2.40). This equation is shown as Eq. (2.41). There is some difficulty in evaluating Eq. (2.41) at $\zeta = 0$ and $0 \le \rho < 1$ since both the numerator and denominator approach zero as ζ approaches zero. Therefore, an expression for the heat generation at $\zeta = 0$ and $0 \le \rho < 1$ was developed using l'Hôpital's rule and is shown as Eq. (2.42).

$$\frac{\partial V}{\partial \rho} = -\frac{4}{\pi}\left(\frac{\frac{1+\rho}{V_1} - \frac{1-\rho}{V_2}}{\left(V_1+V_2\right)^2 \sqrt{1-\frac{4}{\left(V_1+V_2\right)^2}}} \right) \tag{2.39}$$

$$\frac{\partial V}{\partial \zeta} = -\frac{4}{\pi}\left(\frac{\frac{\zeta}{V_1} + \frac{\zeta}{V_2}}{\left(V_1+V_2\right)^2 \sqrt{1-\frac{4}{\left(V_1+V_2\right)^2}}} \right) \tag{2.40}$$

$$Q(\rho,\zeta) = \frac{16\sigma v_0^2}{\pi^2 k T_0}\left[\frac{\left(\frac{1+\rho}{V_1} - \frac{1-\rho}{V_2}\right)^2 + \left(\frac{\zeta}{V_1} + \frac{\zeta}{V_2}\right)^2}{\left(V_1+V_2\right)^4 \left(1-\frac{4}{\left(V_1+V_2\right)^2}\right)} \right]$$

where: (2.41)

$$V_1 = \sqrt{\zeta^2 + (1+\rho)^2} \quad \textit{and} \quad V_2 = \sqrt{\zeta^2 + (1-\rho)^2}$$

$$Q(\rho) = \frac{2\sigma v_0^2}{\pi^2 k T_0}\left(\frac{1}{1+\rho} + \frac{1}{1-\rho} \right) \qquad (\textit{valid for } \zeta = 0 \textit{ and } 0 \le \rho < 1) \tag{2.42}$$

Figure 2.3 shows a contour plot of the dimensionless heat generation. The heat generation is highest near the perimeter of the electrode where a discontinuity in the slope of the electric potential distribution exists. The result is that, at $\rho = 1$ and $\zeta = 0$, the partial derivative with respect to ρ, which is required to evaluate Eq. (2.19), is undefined. From Eqs. (2.41) and (2.42) it can be seen that this discontinuity results in a heat

generation function that approaches infinity at the edge of the electrode. This singularity is a consequence of assuming the radius of curvature along the perimeter of the electrode tip is zero. This assumption is inherent in modeling the tip of the electrode as an ideal flat disk.

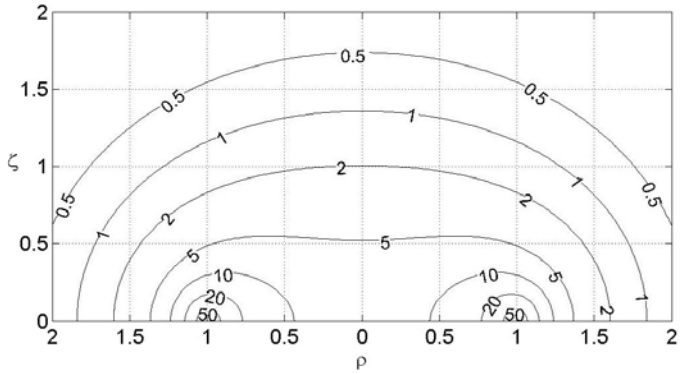

Figure 2.3 Contour plot showing the dimensionless heat generation near the electrode surface.

As will be discussed in Chapter 3, this singularity creates some difficulty in evaluating the solution to the bioheat equation (discussed in Section 2.4, below). Section 3.3 describes the steps taken in handling this singularity to obtain accurate temperature profiles.

2.4 Analytical Solution to the Bioheat Equation

With an expression for the dimensionless heat generation, it is possible to obtain a solution to the bioheat equation. This consists of reducing Eq. (2.7) to an ODE, as described previously, by applying the Hankel and Fourier transforms to remove dependence of θ on ρ and ζ, respectively. Application of the zero-order Hankel transform to Eq. (2.7) results in the following:

$$\int_0^\infty \frac{\partial}{\partial\rho}\left(\rho\frac{\partial\theta}{\partial\rho}\right)J_0(\gamma\rho)\,d\rho+\int_0^\infty \rho\frac{\partial^2\theta}{\partial\zeta^2}J_0(\gamma\rho)d\rho+\int_0^\infty \rho Q(\rho,\zeta)J_0(\gamma\rho)\,d\rho = \int_0^\infty \rho\frac{\partial\theta}{\partial\tau}J_0(\gamma\rho)\,d\rho \tag{2.43}$$

The second term on the left-hand side and the term on the right-hand side of Eq. (2.43) can both be evaluated by reversing the order of operations and using the definition of the Hankel transform to obtain the following results:

$$\int_0^\infty \rho\frac{\partial^2\theta}{\partial\zeta^2}J_0(\gamma\rho)\,d\rho=\frac{\partial^2}{\partial\zeta^2}\int_0^\infty \rho\theta J_0(\gamma\rho)\,d\rho=\frac{\partial^2\bar{\theta}}{\partial\zeta^2} \tag{2.44}$$

$$\int_0^\infty \rho\frac{\partial\theta}{\partial\tau}J_0(\gamma\rho)\,d\rho=\frac{\partial}{\partial\tau}\int_0^\infty \rho\theta J_0(\gamma\rho)\,d\rho=\frac{\partial\bar{\theta}}{\partial\tau} \tag{2.45}$$

Also, the third term on the left-hand side of Eq. (2.43) is simply the definition of the Hankel transform of the dimensionless heat generation function, which can be represented as $\bar{Q}$. The first term on the left-hand side of the equation is, therefore, the only one that remains to be evaluated. This term can be evaluated using integration by parts and the boundary conditions in ρ. Integration by parts results in Eq. (2.46). Using the boundary conditions and the properties of Bessel functions shown in Eq. (2.47), as well as the definition of the Hankel transform, the final result of integration by parts is Eq. (2.48).

$$\int_0^\infty \frac{\partial}{\partial\rho}\left(\rho\frac{\partial\theta}{\partial\rho}\right)J_0(\gamma\rho)\,d\rho = \left[\rho\frac{\partial\theta}{\partial\rho}J_0(\gamma\rho)\right]_0^\infty+\gamma\left[\rho\theta J_1(\gamma\rho)\right]_0^\infty-\gamma^2\int_0^\infty \rho\theta J_0(\gamma\rho)dp \tag{2.46}$$

$$\left.\frac{\partial\theta}{\partial\rho}\right|_{\rho=0}=\left.\frac{\partial\theta}{\partial\rho}\right|_{\rho\to\infty}=\theta(\rho,\zeta,\tau)_{\rho\to\infty}=J_0(\gamma\rho)_{\rho\to\infty}=J_1(0)=J_1(\gamma\rho)_{\rho\to\infty}=0 \tag{2.47}$$

$$\int_0^\infty \frac{\partial}{\partial \rho}\left(\rho \frac{\partial \theta}{\partial \rho}\right) J_0(\gamma\rho)\, d\rho = -\gamma^2 \bar{\theta} \tag{2.48}$$

Thus, from Eqs. (2.44), (2.45), and (2.48) the result of applying the Hankel transform to the bioheat equation is the following:

$$-\gamma^2 \bar{\theta} + \frac{\partial^2 \bar{\theta}}{\partial \zeta^2} + \bar{Q} = \frac{\partial \bar{\theta}}{\partial \tau} \tag{2.49}$$

To reduce Eq. (2.49) to an ODE, the Fourier transform may now be applied to remove dependence on ζ. Doing this, the following equation is obtained:

$$\begin{aligned} -\gamma^2 \int_0^\infty K(\beta,\zeta)\bar{\theta}\, d\zeta + \int_0^\infty K(\beta,\zeta)\frac{\partial^2 \bar{\theta}}{\partial \zeta^2} d\zeta + \int_0^\infty K(\beta,\zeta)\bar{Q} d\zeta \\ = \int_0^\infty K(\beta,\zeta)\frac{\partial \bar{\theta}}{\partial \tau} d\zeta \end{aligned} \tag{2.50}$$

The first and last terms on the left-hand side may be evaluated using the definition of the Fourier transform to obtain Eqs. (2.51) and (2.52), respectively. The term on the right-hand side of Eq. (2.50) may be evaluated by changing the order of operations and, then, using the definition of the Fourier transform to obtain Eq. (2.53).

$$-\gamma^2 \int_0^\infty K(\beta,\zeta)\bar{\theta}\, d\zeta = -\gamma^2 \bar{\bar{\theta}} \tag{2.51}$$

$$\int_0^\infty K(\beta,\zeta)\bar{Q} d\zeta = \bar{\bar{Q}} \tag{2.52}$$

$$\int_0^\infty K(\beta,\zeta)\frac{\partial \bar{\theta}}{\partial \tau} d\zeta = \frac{\partial}{\partial \tau}\int_0^\infty K(\beta,\zeta)\bar{\theta}\, d\zeta = \frac{\partial \bar{\bar{\theta}}}{\partial \tau} \tag{2.53}$$

The second term on the left-hand side of Eq. (2.50) can be integrated by parts. The following expression is, thus, obtained:

$$\left[K(\beta,\zeta)\frac{\partial \bar{\theta}}{\partial \zeta}\right]_0^\infty - \left[K'(\beta,\zeta)\bar{\theta}\right]_0^\infty + \int_0^\infty K''(\beta,\zeta)\bar{\theta}\, d\zeta \tag{2.54}$$

The first and second derivatives of the kernel for the Fourier transform are shown below:

$$K'(\beta,\zeta)=\beta\sqrt{\frac{2}{\pi}}\left[\frac{-\beta\sin\beta\zeta+Bi\cos\beta\zeta}{\sqrt{\beta^2+Bi^2}}\right] \tag{2.55}$$

$$K''(\beta,\zeta)=-\beta^2\sqrt{\frac{2}{\pi}}\left[\frac{\beta\cos\beta\zeta+Bi\sin\beta\zeta}{\sqrt{\beta^2+Bi^2}}\right] \tag{2.56}$$

Noting from Eqs. (2.22) and (2.56) that $K''(\beta,\zeta)=-\beta^2K(\beta,\zeta)$ and using, once again, the definition of the Fourier transform, the third term in Eq. (2.54) may be evaluated to obtain the result shown in Eq. (2.57).

$$\int_0^\infty K''(\beta,\zeta)\bar{\theta}\,d\zeta=-\beta^2\int_0^\infty K(\beta,\zeta)\bar{\theta}\,d\zeta=-\beta^2\bar{\bar{\theta}} \tag{2.57}$$

To evaluate the first two terms in Eq. (2.54), the Hankel transform of the ζ boundary conditions shown in Eq. (2.10) must be evaluated. Applying the Hankel transform, the following expressions are obtained:

$$\int_0^\infty \rho\left.\frac{\partial\theta}{\partial\zeta}\right|_{\zeta=0} J_0(\gamma\rho)\,d\rho=Bi\int_0^\infty \rho\theta(\rho,0,\tau)J_0(\gamma\rho)\,d\rho \tag{2.58}$$

$$\int_0^\infty \rho\theta(\rho,\zeta,\tau)_{\zeta\to\infty} J_0(\gamma\rho)\,d\rho=0 \tag{2.59}$$

Again, using the definition of the Hankel transform and a change in the order of operations for the term on the left-hand side of Eq. (2.58), the following equations are obtained:

$$\left.\frac{\partial\bar{\theta}}{\partial\zeta}\right|_{\zeta=0}=Bi\bar{\theta}(\gamma,0,\tau) \tag{2.60}$$

$$\bar{\theta}(\gamma,\zeta,\tau)_{\zeta\to\infty}=0 \tag{2.61}$$

Using the results shown in Eqs. (2.57), (2.60) and (2.61), and recognizing that K and K' are always finite and that $\partial\bar{\theta}/\partial\zeta$ must approach zero as ζ approaches infinity, the following is obtained from Eq. (2.54):

$$-\left[K(\beta,\zeta)\,Bi\bar{\theta}\right]_{\zeta=0}+\left[K'(\beta,\zeta)\,\bar{\theta}\right]_{\zeta=0}-\beta^2\bar{\bar{\theta}} \tag{2.62}$$

The final result of integration by parts is obtained by evaluating the necessary terms in Eq. (2.62) at $\zeta=0$ to obtain the following:

$$\begin{aligned}\int_0^\infty K(\beta,\zeta)\frac{\partial^2\bar{\theta}}{\partial\zeta^2}d\zeta &= -K(\beta,0)\,Bi\bar{\theta}_{\zeta=0}+K'(\beta,0)\bar{\theta}_{\zeta=0}-\beta^2\bar{\bar{\theta}}\\ &= -\beta\sqrt{\frac{2}{\pi\left(\beta^2+Bi^2\right)}}Bi\bar{\theta}_{\zeta=0}+Bi\beta\sqrt{\frac{2}{\pi\left(\beta^2+Bi^2\right)}}\bar{\theta}_{\zeta=0}-\beta^2\bar{\bar{\theta}}\\ &= -\beta^2\bar{\bar{\theta}}\end{aligned} \tag{2.63}$$

From Eqs. (2.51) - (2.53) and Eq. (2.63), the following is obtained as a result of applying both the Hankel and Fourier transforms to the bioheat equation:

$$\frac{d\bar{\bar{\theta}}}{d\tau}+\left(\gamma^2+\beta^2\right)\bar{\bar{\theta}}=\bar{\bar{Q}} \tag{2.64}$$

This equation is a first-order, linear, non-homogeneous ODE that may be solved using an appropriate integrating factor. Using this method, the following results are obtained:

$$\frac{d\bar{\bar{\theta}}}{d\tau}e^{(\gamma^2+\beta^2)\tau}+(\gamma^2+\beta^2)e^{(\gamma^2+\beta^2)\tau}\bar{\bar{\theta}}=\frac{d}{d\tau}\left[\bar{\bar{\theta}}e^{(\gamma^2+\beta^2)\tau}\right]=\bar{\bar{Q}}e^{(\gamma^2+\beta^2)\tau} \tag{2.65}$$

Integration with respect to τ and division by $e^{(\gamma^2+\beta^2)\tau}$ results in Eq. (2.66) where C is a constant of integration.

$$\bar{\bar{\theta}}=\frac{\bar{\bar{Q}}}{\gamma^2+\beta^2}+Ce^{-\left(\gamma^2+\beta^2\right)\tau} \tag{2.66}$$

Equation (2.67) is obtained by taking the Hankel and Fourier transforms of the initial condition in Eq. (2.8). This transformed initial condition may be used to evaluate the constant of integration in Eq. (2.66). The final solution to Eq. (2.64) is shown in Eq. (2.68).

$$\bar{\bar{\theta}}(0) = 0 \tag{2.67}$$

$$\bar{\bar{\theta}} = \frac{\bar{\bar{Q}}\left[1 - e^{-\left(\gamma^2+\beta^2\right)\tau}\right]}{\gamma^2 + \beta^2} \tag{2.68}$$

The final solution to the bioheat equation is, then, obtained by applying the inverse Hankel and Fourier transforms to Eq. (2.68). The complete solution is shown in Eqs. (2.69) and (2.70) below:

$$\theta\left(\rho,\zeta,\tau\right) = \int_0^\infty \int_0^\infty \gamma K\left(\beta,\zeta\right) \frac{\bar{\bar{Q}}\left[1 - e^{-\left(\gamma^2+\beta^2\right)\tau}\right]}{\gamma^2 + \beta^2} J_0\left(\gamma\rho\right) d\beta d\gamma \tag{2.69}$$

$$\bar{\bar{Q}} = \int_0^\infty \int_0^\infty \rho J_0\left(\gamma\rho\right) K\left(\beta,\zeta\right) Q\left(\rho,\zeta\right) d\rho d\zeta \tag{2.70}$$

CHAPTER 3 – EVALUATION OF THE ANALYTICAL SOLUTION

3.1 Early Attempts

As seen in Eqs. (2.69) and (2.70), the solution to the bioheat equation obtained using integral transforms has the form of a surface integral (resulting from the forward transforms of the dimensionless heat generation function) nested within another surface integral (resulting from application of the inverse transforms). Because of the complexity of the solution, evaluation using analytical integration techniques was not possible. Instead, numerical methods of evaluation were used. Early attempts at evaluating Eqs. (2.69) and (2.70) were made using two commercial software packages, Maple® and Mathcad®. Because of difficulties encountered in evaluating the analytical solution to the bioheat equation and limitations of these software packages, it was necessary to write a C program to evaluate the solution. This allowed much more flexibility in the evaluation methods than was possible using commercial software packages. Sections 3.1.1 – 3.1.3 describe the various attempts made in evaluating Eqs. (2.69) and (2.70) and what was learned from these early attempts. The motivation for doing so is to offer a justification for choosing the method described in Sections 3.2 – 3.5 and to show why this method is believed to be superior to those used in earlier attempts.

3.1.1 Commercial Software Packages

Before discovering an analytical solution to the integral in Eq. (2.36), simply evaluating and generating a plot of the heat generation function was very difficult. Using

Eq. (2.19) to develop a function from Eq. (2.36) for the heat generation results in an expression with two integrals similar to that in Eq. (2.36). These integrals have highly oscillatory integrands making it particularly challenging for Maple to evaluate them [13]. Only Mathcad was successful in generating plots of the heat generation function, which it did in only a few seconds. However, because numerical evaluation of Eqs. (2.69) and (2.70) would require a very large number of function evaluations of the heat generation, the computation time required by Mathcad to generate a heat generation plot was considered to be too long if temperature profiles were to be generated within a reasonable amount of time. Therefore, discovering an analytical solution to the integral in Eq. (2.36) was an important step in evaluating the solution to the bioheat equation. Two reasons for this are, first, it reduced the error associated with numerical evaluation and, second, it reduced computation time. Not only was the level of error reduced to machine error as opposed to the relatively large error associated with numerical integration, but the cumbersome and time consuming calculations required by this method were eliminated.

Numerical accuracy and computation time were both very important considerations, especially at the early stages of evaluation. The level of numerical error associated with evaluating the heat generation was a concern since error at the early stages was expected to propagate and increase through the later stages of evaluation. Obtaining accurate temperature fields, therefore, required minimizing error, particularly in the early stages. Likewise, reduction of computation time was especially important at the early stages since the time required to evaluate subsequent integrals was expected to increase exponentially.

To investigate the feasibility of using a commercial software package to evaluate the forward and inverse integral transforms, the Fourier and Hankel transforms of the heat generation were evaluated individually. Before an analytical solution to Eq. (2.36) was found, only Mathcad was successful in evaluating the Fourier transform and neither software package was able to evaluate the Hankel transform. With the analytical solution, both Mathcad and Maple successfully generated plots of the Fourier transform in only a few seconds. However, only Mathcad could evaluate the Hankel transform. Furthermore, while attempts to evaluate the Hankel transform of the Fourier-transformed heat generation function (or vice versa) were successful using Mathcad, over an hour was required to generate a surface plot of 400 points with the default precision. Because numerical evaluation of the inverse transforms to obtain temperature profiles would require several more evaluations of the forward transforms than are needed to make a surface plot, this was determined to be too computationally expensive. Furthermore, certain measures had to be taken to avoid evaluating the heat generation at $\rho = 1$ and $\zeta = 0$ during the process of evaluating the Hankel and Fourier transforms. As mentioned previously, a singularity exists in the heat generation function at this point. One approach taken to avoid this problem was to change the lower limit on the Fourier transform to some small, but non-zero value. However, such measures introduced an indeterminate amount of error into the results.

3.1.2 Numerical Integration Routine in C

It was eventually decided that the best approach would be to write a program to evaluate the solution to the bioheat equation. This would provide much more control over and flexibility of the method of evaluation. For example, difficulties encountered

while using commercial software, such as the inability to handle integration of the singularity at $\rho = 1$ and $\zeta = 0$, would be more manageable because a program could be tailored to this specific problem.

A routine that uses the trapezoidal method of integration was written in C. This routine made use of three functions (*trapzd*, *qromb*, and *polint*) from *Numerical Recipes in C: The Art of Scientific Computing* [14]. The general method of evaluation consisted of obtaining increasingly accurate estimates for a given integral by dividing the area between the limits of integration into progressively smaller trapezoidal sections and summing the areas. The first estimate is simply the area of one trapezoid, the second, the area of two trapezoids, the third, the area of four, and so on. This is illustrated in Fig. 3.1.

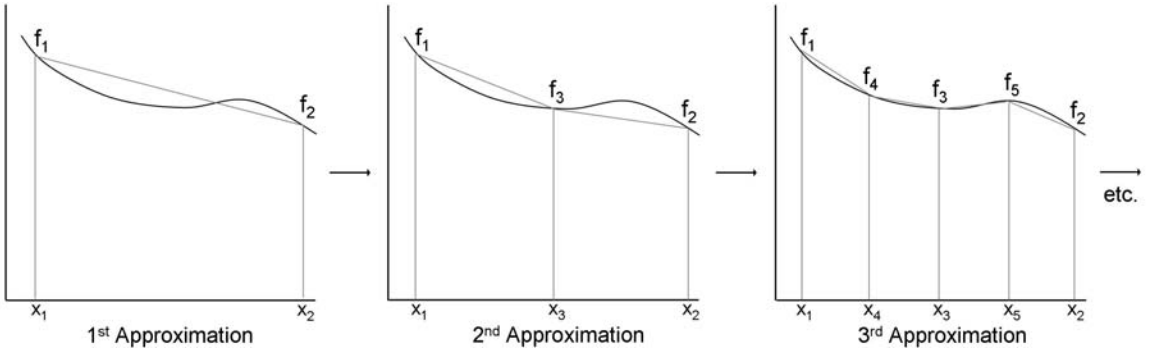

Figure 3.1 Schematic of numerical integration using the trapezoidal method.

The C function, *trapzd*, is written such that, once the function to be integrated has been evaluated at a given location, it does not need to be evaluated again in subsequent approximations. In other words, in Fig. 3.1, f_1 and f_2 do not need to be recalculated for the second approximation and f_1, f_2, and f_3, likewise, do not need to be recalculated for the third approximation. Equations (3.1) - (3.4) illustrate this concept, where A_1, A_2, and A_3, represent the first, second, and third approximations, respectively, of the total area

between the limits of integration, x_1 and x_2. The notation, $\Delta x_{n,m}$, is defined as $x_n - x_m$. This derivation is based upon the assumption that the area is divided in half with each iteration.

$$A_1 = \Delta x_{2,1}\left(\frac{f_1 + f_2}{2}\right) = \Delta x_{2,1} S_1 \tag{3.1}$$

$$\begin{aligned} A_2 &= \Delta x_{3,1}\left(\frac{f_1 + f_3}{2}\right) + \Delta x_{2,3}\left(\frac{f_3 + f_2}{2}\right) \\ &= \frac{\Delta x_{2,1}}{2}\left(\frac{f_1 + f_2}{2} + f_3\right) = \frac{\Delta x_{2,1}}{2}\left(S_1 + f_3\right) = \frac{\Delta x_{2,1}}{2} S_2 \end{aligned} \tag{3.2}$$

$$\begin{aligned} A_3 &= \Delta x_{4,1}\left(\frac{f_1 + f_4}{2}\right) + \Delta x_{3,4}\left(\frac{f_4 + f_3}{2}\right) + \Delta x_{5,3}\left(\frac{f_3 + f_5}{2}\right) + \Delta x_{2,5}\left(\frac{f_5 + f_2}{2}\right) \\ &= \frac{\Delta x_{2,1}}{4}\left(\frac{f_1 + f_2}{2} + f_3 + f_4 + f_5\right) = \frac{\Delta x_{2,1}}{4}\left(S_2 + f_4 + f_5\right) = \frac{\Delta x_{2,1}}{4} S_3 \end{aligned} \tag{3.3}$$

Equations (3.1) - (3.3) lead to the following general formula for calculating the n^{th} approximation for the integral of an arbitrary function of x, $f(x)$, between the limits, x_1 and x_2.

$$\begin{aligned} A_n &= \frac{\Delta x_{2,1}}{2^{n-1}} S_n \\ &where: \\ S_1 &= \frac{f_1 + f_2}{2} \\ S_n &= S_{n-1} + \sum f_{new} \quad for\ n > 1 \end{aligned} \tag{3.4}$$

The summation in Eq. (3.4) represents the sum of all new function evaluations needed for the n^{th} approximation that were not obtained in previous iterations. The total number of function evaluations that would be required without using previously calculated values is $2^{n-1}+1$ while the number of terms in this summation is equal to 2^{n-2}. Therefore, the

number of function evaluations for the n^{th} approximation where $n > 1$ is reduced by $2^{n-2}+1$.

The C function, *trapzd*, uses the results shown in Eq. (3.4) to calculate an approximate area for any specified function, set of integration limits, and *n* value. The *trapzd* function is called by *qromb*, which passes in the necessary parameters and monitors convergence as it increments *n*. To speed convergence, *qromb* uses the routine, *polint*, to fit an n^{th}-order polynomial to the set of approximate areas already calculated and, from this polynomial, to predict the actual area to which the approximate values are converging. The solution is considered converged when the predicted area changes by less than a specified percent from the previous prediction.

3.1.3 Reformulation of the Fourier and Hankel Transforms

With the integration routine described in Section 3.1.2, attempts were again made at evaluating the Fourier and Hankel transforms. Difficulties were still encountered mainly because of the infinite upper limit of integration and the oscillatory nature of the integrand for both transforms. A literature search on Fourier and Hankel transforms revealed that there are alternate forms into which such integrals may be cast to eliminate these difficulties [15-16]. One such method consists of converting Fourier sine and cosine integrals into infinite summations of integrals with finite limits of integration. These integrals are much easier to calculate because of the finite limits of integration as well as the fact that the integrands are not oscillatory over the range of integration. Furthermore, convergence of the summation can be easily accomplished by monitoring when the magnitude of the integrals drop below a specified value. It was, also, found that

a Hankel transform can be recast into a form containing a Fourier transform. Therefore, the Hankel transform could simply be evaluated using the same methods.

Although, with this approach, the forward transforms of the heat generation function could be evaluated in less time than that required by Mathcad, the computation time was too long for such an approach to be used in generating temperature profiles. In an attempt to further reduce computation time, a fast Fourier transform routine from *Numerical Recipes in C* was used instead of the integration routine described previously. However, both the computation time and accuracy of this routine were comparable to the integration routine.

3.2 Development of an Alternate Approach

While the computation time associated with these approaches is adequate for evaluating the forward transforms of the heat generation function, it is far from adequate for the subsequent evaluation of the inverse transforms. The fundamental weakness of this approach is that it requires the evaluation of three nested integrals, which is very inefficient. This becomes apparent by considering a hypothetical situation in which each integral requires on the order of 10^3 evaluations of its integrand to converge to a solution (which, for a highly oscillatory integrand integrated over a range that approximates a semi-infinite range of integration, is a reasonable estimate). If evaluation of the inner-most integral requires 1 ms, it would require 10^6 seconds, or about 11.5 days, to evaluate the outer-most integral only once. To generate a temperature profile containing 100 data points would require over three years. It is apparent from this example that the time required to evaluate Eqs. (2.69) and (2.70) can be significantly reduced by minimizing how often an integral must be evaluated inside another integral. It was, therefore,

determined that an approach fundamentally different from those described previously would be necessary.

It was recognized that the number of nested integrals could be reduced if a method of evaluating surface integrals was developed. If so, evaluation of Eqs. (2.69) and (2.70) would simply consist of evaluating one surface integral nested within another. Although, surface integrals would require more time to calculate, only one of the two would require evaluation of another as part of its integrand. This proved to be a much better approach.

3.2.1 Description of a Surface Integration Routine

A method of evaluating surface integrals was developed that is analogous to the previously discussed trapezoidal method used for integration with respect to only one variable. Instead of approximating the area under a curve, this method consists of approximating the volume under a surface whose boundaries are specified by the upper and lower limits on the integrals. An iterative process, again, may be used in which the surface section to be integrated is divided into progressively smaller subsections whose individual volumes are summed to give an approximation for the total volume. This is illustrated in Fig. 3.2, which shows how integration of an arbitrary function, $f(\gamma,\beta)$, between finite limits of integration, may be accomplished. In the present case, this function represents the integrand of Eq. (2.69). Again, the polynomial interpolation routine, *polint*, may be used to reduce the time to convergence by predicting actual volumes based on approximate values calculated from each iteration. Each prediction is compared with that from the previous iteration until the difference between the two is less than a specified value.

A derivation similar to that shown in Eqs. (3.1) - (3.4) may be used to obtain a generalized equation for use in a surface integration routine that does not require the recalculation of previously calculated values. To emphasize this fact, the schematic for each approximation illustrated in Fig. 3.2 shows in bold the new points at which the integrand must be evaluated. For example, to obtain the third approximation, the integrand only needs to be evaluated at points 10 through 25, since the integrand was already calculated at points 1 through 9 in obtaining first and second approximations.

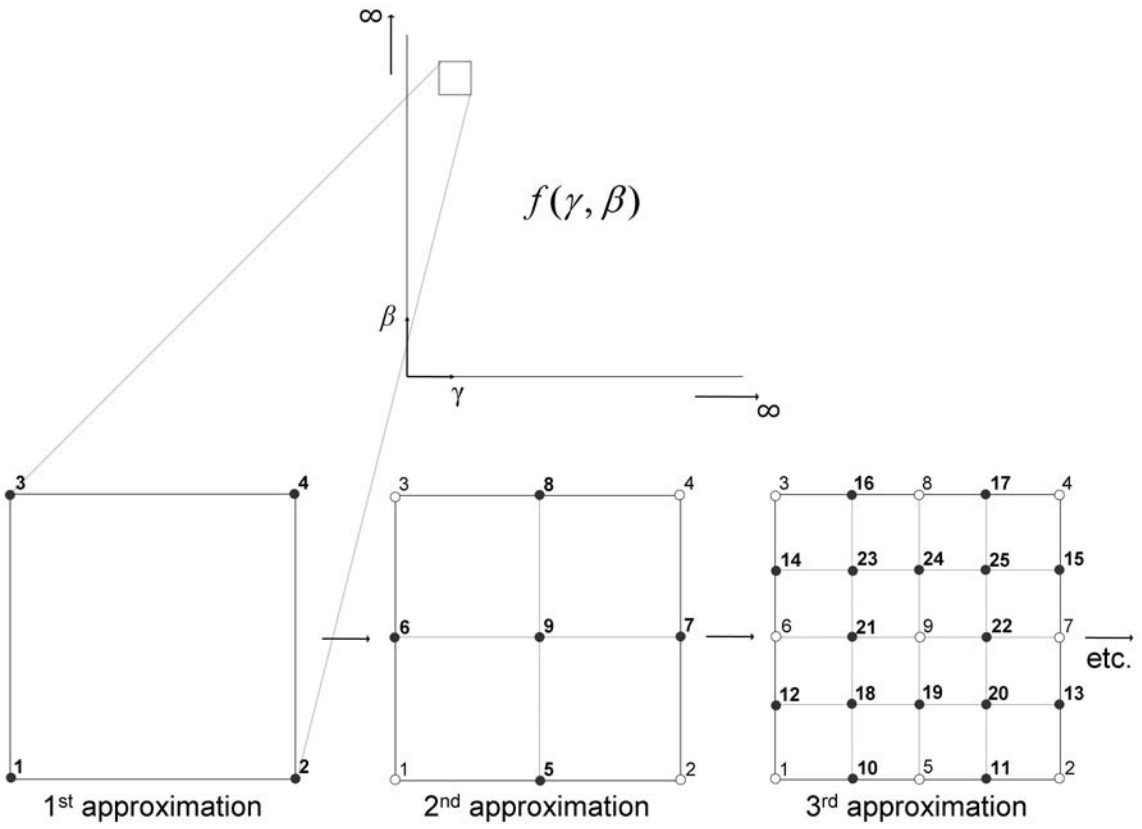

Figure 3.2 Schematic illustrating how a surface integral may be numerically evaluated.

While the reduction in computation time for the first few approximations is very small, it becomes particularly significant in subsequent iterations since the number of volumes to calculate increases by a factor of four with each iteration.

The derivation of the necessary equations for use in the surface integration routine just described is shown in Eqs. (3.5) - (3.8), where V_1, V_2, and V_3, represent the first, second, and third approximations, respectively, of the total volume between the limits of integration, γ_1 and γ_2 and β_1 and β_3 ($\gamma_1 = \gamma_3$ and $\beta_1 = \beta_2$ as seen in Fig. 3.2). The notation, $\Delta\gamma_{n,m}$, signifies the quantity, $\gamma_n - \gamma_m$. Likewise, $\Delta\beta_{n,m} \equiv \beta_n - \beta_m$. Also, f_n is defined as $f(\gamma_n, \beta_n)$.

$$V_1 = \Delta\gamma_{2,1}\Delta\beta_{3,1}\left(\frac{f_1 + f_2 + f_3 + f_4}{4}\right) = \Delta\gamma_{2,1}\Delta\beta_{3,1}S_1 \tag{3.5}$$

$$\begin{aligned} V_2 &= \Delta\gamma_{5,1}\Delta\beta_{6,1}\left(\frac{f_1 + f_5 + f_6 + f_9}{4}\right) + \Delta\gamma_{2,5}\Delta\beta_{9,5}\left(\frac{f_5 + f_2 + f_9 + f_7}{4}\right) \\ &\quad + \Delta\gamma_{9,6}\Delta\beta_{3,6}\left(\frac{f_6 + f_9 + f_3 + f_8}{4}\right) + \Delta\gamma_{7,9}\Delta\beta_{8,9}\left(\frac{f_9 + f_7 + f_8 + f_4}{4}\right) \\ &= \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{4}\left(\frac{f_1 + f_2 + f_3 + f_4}{4} + \frac{f_5 + f_6 + f_7 + f_8}{2} + f_9\right) \\ &= \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{4}\left(S_1 + \frac{f_5 + f_6 + f_7 + f_8}{2} + f_9\right) = \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{4}S_2 \end{aligned} \tag{3.6}$$

$$\begin{aligned} V_3 &= \Delta\gamma_{10,1}\Delta\beta_{12,1}\left(\frac{f_1 + f_{10} + f_{12} + f_{18}}{4}\right) + \Delta\gamma_{5,10}\Delta\beta_{18,10}\left(\frac{f_{10} + f_5 + f_{18} + f_{19}}{4}\right) \\ &\quad + \ldots \\ &\quad + \Delta\gamma_{25,24}\Delta\beta_{8,24}\left(\frac{f_{24} + f_{25} + f_8 + f_{17}}{4}\right) + \Delta\gamma_{15,25}\Delta\beta_{17,25}\left(\frac{f_{25} + f_{15} + f_{17} + f_4}{4}\right) \\ &= \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{16}\left(S_2 + \frac{f_{10} + f_{11} + \ldots + f_{17}}{2} + f_{18} + f_{19} + \ldots + f_{25}\right) \\ &= \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{16}\left(S_2 + \frac{\sum f_{side,new}}{2} + \sum f_{interior,new}\right) = \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{16}S_3 \end{aligned} \tag{3.7}$$

With each new approximation of the total volume over the range of integration, the surface subsections are each divided into four new subsections. This results in new points, both inside of and along the edges of the region over which the surface is being integrated. Therefore, in Eqs. (3.7) and (3.8), $f_{interior,new}$ represents any new point inside the region for which the integrand has not been calculated previously. Likewise, $f_{side,new}$ represents any new point along the boundary of the region for which the integrand has not been calculated previously.

Equations (3.5) - (3.7) lead to the general formula in Eq. (3.8) for calculating the n^{th} approximation for the surface integral of an arbitrary function, $f(\gamma,\beta)$, between the limits of integration, γ_1 and γ_2 and β_1 and β_3.

$$
\begin{aligned}
&V_n = \frac{\Delta\gamma_{2,1}\Delta\beta_{3,1}}{4^{n-1}} S_n \\
&where: \\
&S_1 = \frac{f_1 + f_2 + f_3 + f_4}{4} \\
&S_n = S_{n-1} + \frac{\sum f_{side,new}}{2} + \sum f_{interior,new} \quad for\ n > 1
\end{aligned}
\tag{3.8}
$$

The total number of function evaluations for any given approximation that would be required without using previously calculated values is $4^{n-1} + 2^n + 1$ while the number of new function evaluations (both inside of and along the edges of the region to be integrated) necessary for the n^{th} approximation is $3 \cdot 4^{n-2} + 2^{n-1}$ (where $n > 1$). Therefore, using the formulation outlined in Eqs. (3.5) - (3.8), the number of function evaluations for the n^{th} approximation, where $n > 1$, is reduced by $4^{n-2} + 2^{n-1} + 1$.

A C function called *prism* that is analogous to *trapzd* was written that uses the result shown in Eq. (3.8) to calculate an approximate volume for any specified function,

set of integration limits, and n value. The *prism* function is called by *qrombPrism*, which, like *qromb*, passes in the necessary parameters and monitors convergence as n is incremented. Again, *polint* is used by *qrombPrism* to speed convergence by fitting an n^{th}-order polynomial to the set of approximate volumes already calculated and extrapolating the actual volume using this polynomial. The solution is considered converged when the calculated volume changes by less than a specified percent from the previous evaluation.

3.3 Implementation of the Surface Integration Routine

The surface integration routine described above was used in evaluating Eqs. (2.69) and (2.70). However, this routine by itself may only be used to evaluate surface integrals with finite limits of integration. Evaluation of Eqs. (2.69) and (2.70) is complicated by the fact that the upper limits of integration are infinite. These surface integrals may be evaluated, however, by integrating over several finite sections, adding the volume of each section to a total volume. This can be done systematically by stepping along the surface, maintaining the lower and upper limits in one dimension constant while the limits in the other dimension are incremented. Thus, surface sections are integrated along a strip or row of the whole surface until the magnitude of the section volumes drop below a specified value. Because the integrands of both Eq. (2.69) and Eq. (2.70) are oscillatory, there is the possibility of satisfying this convergence criterion before the integrand has damped out sufficiently. This occurs if the magnitude of the negative volume for a given surface section is close to that of the positive volume. In such a case, the magnitude of the total volume may be small enough to satisfy the convergence criterion even if the amplitude of the oscillatory integrand is still large.

Therefore, an additional condition for convergence was used that required two consecutive section volumes to be below the specified value for convergence. Once this additional condition is satisfied, the upper and lower limits that were previously held constant are incremented and the same process repeated to find the volume along the second row. In this manner, the total volume may be calculated by summing row volumes until the total volume changes by less than a specified value. This process is illustrated in Fig. 3.3. Row volumes are obtained by summing section volumes along the row while the total volume is obtained by summing the row volumes.

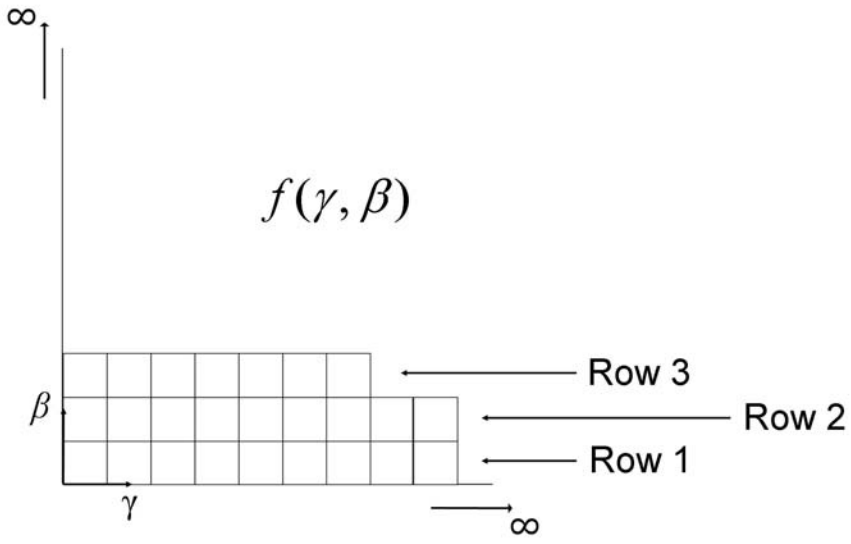

Figure 3.3 Schematic illustrating the process by which surface integrals are evaluated. Row volumes are obtained by summing section volumes along the row while the total volume is obtained by summing the row volumes.

This surface integration routine was validated by evaluating an integral of the same form as Eq. (2.70) but with the heat generation function replaced by a simpler function that was chosen such that Eq. (2.70) would have a known analytical solution.

This function is shown in Eq. (3.9). Like the actual heat generation function, this function approaches zero as ρ and ζ approach infinity.

$$Q^*(\rho,\zeta) = de^{-b\zeta} \Big/ \left(\rho^2 + a^2\right) \tag{3.9}$$

Values of 0.2, 2, and 20 were assigned to b, a, and d, respectively, while the Biot number was assigned a value of 8. The values for a and d were chosen to scale Eq. (3.9) such that the values obtained from evaluating Eq. (2.70) would be of similar order to the values obtained when using the actual heat generation function. The value of b governs the rate at which Eq. (3.9) decays as ζ increases and was chosen such that evaluation of Eq. (2.70) would require integration over a range at least as large as that required when evaluating Eq. (2.70) using the actual heat generation. Finally, the value for the Biot number was chosen such that the corresponding convection coefficient obtained when R = 1.3 mm and k = 0.531 W/m·K would be within the range of values for which temperature profiles were generated. Table 1 compares the program output to the exact solution for several values of γ and β. These results show that the surface integration routine used in this study is very accurate.

A difficulty arises when attempting to evaluate Eq. (2.70) using the actual heat generation function. As mentioned previously, a discontinuity exists in the slope of the electric potential distribution at the electrode tip so that the partial derivative with respect to the radial variable required to evaluate Eq. (2.19) is undefined. This discontinuity results in a heat generation function that approaches infinity as the values of ρ and ζ approach 1 and 0, respectively. This makes integration of the surface section containing this singularity challenging.

Table 3.1 Validation of numerical integration by comparison with an exact solution.

γ	β	Program	Exact	% Error
0.1	0.1	57.33214	57.33219	0.00009
0.6	0.1	10.41781	10.41863	0.00789
0.2	0.3	42.03906	42.03955	0.00116
0.7	0.3	9.18975	9.19056	0.00879
0.3	0.5	21.88426	21.88427	0.00003
0.8	0.5	5.28966	5.29021	0.01042
0.4	0.7	12.16668	12.16676	0.00066
0.9	0.7	3.14023	3.14057	0.01072
0.5	0.9	7.24593	7.24592	0.00005
1	0.9	1.95988	1.96014	0.01321

To make integration of this surface section more manageable, the heat generation function was truncated in the vicinity of the singularity. A maximum dimensionless heat generation rate, QMAX, was specified so that whenever the program attempted to evaluate the heat generation at $\rho = 1$ and $\zeta = 0$, it was automatically assigned the value of QMAX. Likewise, if the value of the heat generation at any other point close to the singularity exceeded QMAX, it was also assigned this value. To find an optimum value for QMAX, Eq. (2.70) was evaluated many times for different values of γ and β, increasing the value of QMAX each time until the results no longer changed appreciably. Figure 3.4 shows the change in Eq. (2.70) with QMAX for nine different sets of values for γ and β. All values were normalized using the initial values obtained for QMAX = 20 so that they would be of the same order. From Fig. 3.4, it can be seen that, above a value of 1000, the result of increasing QMAX is minimal. In fact, for all nine cases, the values increase by less than 0.5 % when QMAX is increased from 1000 to 2000. A value of 2000 was, therefore, used in generating all temperature profiles.

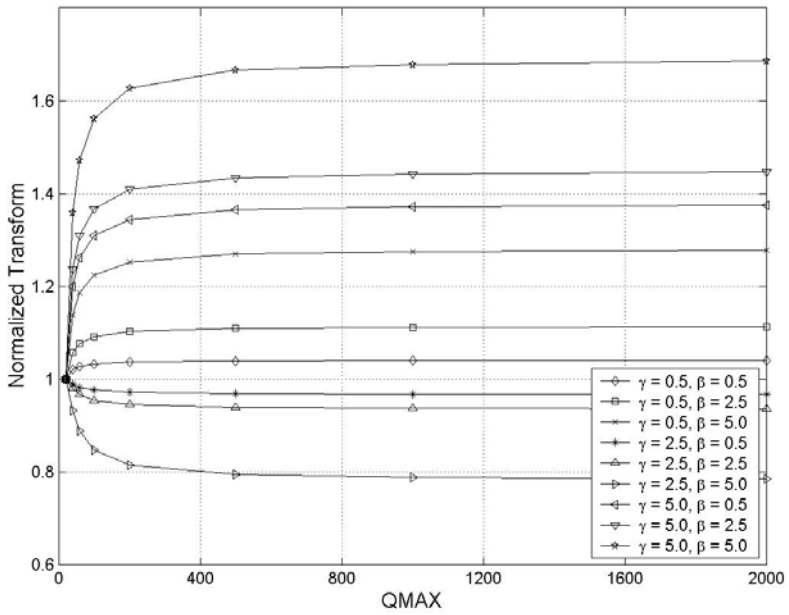

Figure 3.4 Results of an optimization study to determine an appropriate value for QMAX. Equation (2.70) (the Hankel and Fourier transforms of $Q(\rho,\zeta)$) was evaluated using different values of QMAX for several different values of γ and β. Results were normalized using the values obtained for QMAX = 20.

The result of assigning a maximum heat generation rate can be seen in Fig. 3.5. This figure shows surface plots of the dimensionless heat generation rate in the vicinity of the singularity with the singularity truncated at a value of 2000. The range over which the heat generation is plotted in Fig. 3.5a is four times that of Fig. 3.5b. It should also be noted that the range over which the heat generation is plotted in Fig. 3.5a represents only a tenth of the electrode radius in each direction from the singularity. This should emphasize that the reduction in volume due to truncation is very minimal.

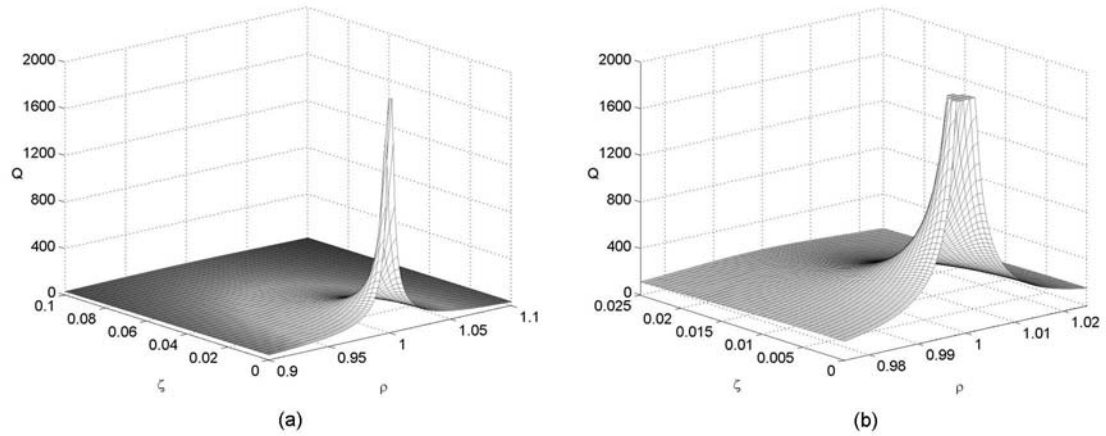

Figure 3.5 Surface plots of the dimensionless heat generation in the immediate vicinity of the truncated singularity (QMAX = 2000). The range over which the heat generation is plotted in (a) is four times that of (b).

3.4 Flow Chart for the Program Used to Generate Temperature Profiles

The previous two sections described the development and validation of a surface integration routine that may be used in evaluating the integrals in Eqs. (2.69) and (2.70). However, some of the details as to how temperature profiles were generated were omitted. A flow chart is shown in Fig. 3.6 that further details the algorithm. The names of C functions are italicized. The following is a detailed description of the steps outlined in Fig. 3.6:

(1) *Load_array_2d* accepts inputs specifying the range of ρ and ζ values for which dimensionless temperatures will be evaluated as well as an integer used to determine the step size between adjacent data points. These parameters are defined as global constants.

(2) With these specified parameters, *Load_array_2d* generates a set of dimensionless temperatures, θ, by calling the function, *Temperature*, for the specified ρ and ζ values. The temperatures are stored in a matrix and, also, written to a data file.

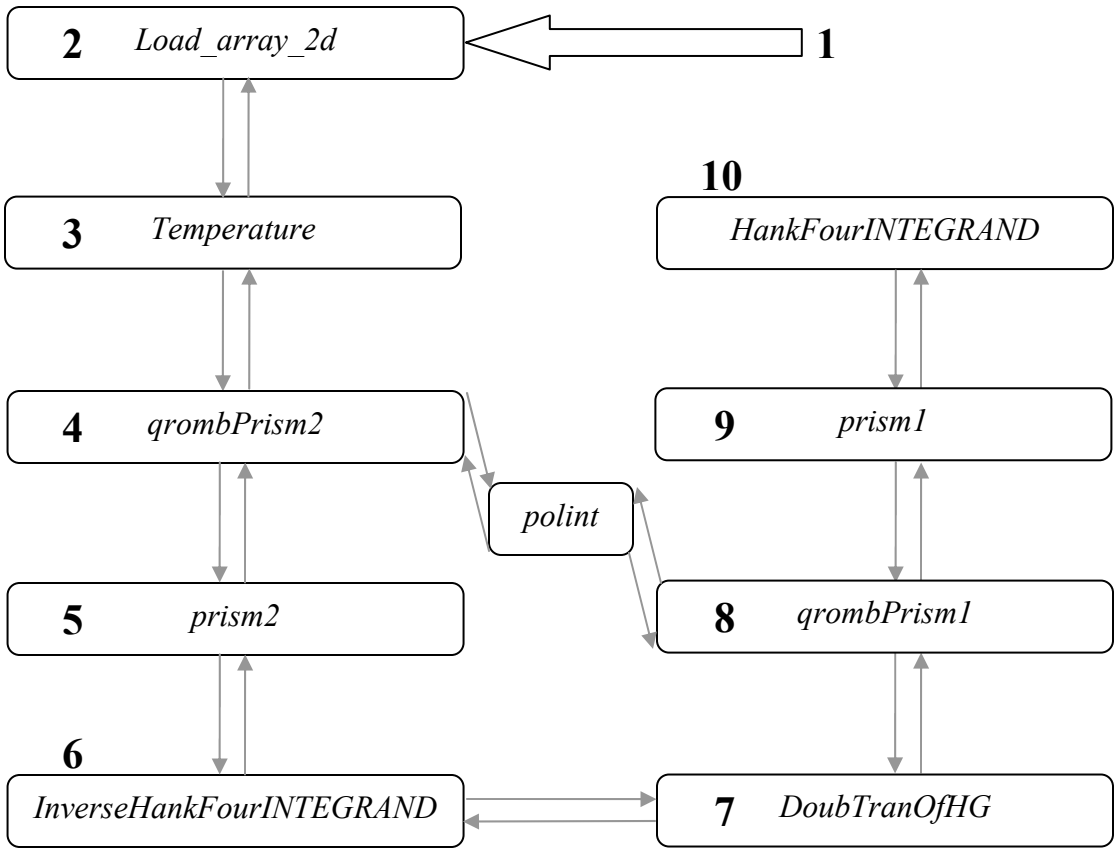

Figure 3.6 A flow chart outlining the sequence of function calls required to generate a dimensionless temperature profile.

(3) The integrand of Eq. (2.69) is shown as Eq. (3.10). To calculate dimensionless temperatures, ρ, ζ, and τ must be substituted into Eq. (3.10) and the resulting expression integrated with respect to γ and β (While values of ρ and ζ are provided by *Load_array_2d*, the value of τ is declared as a global constant). As described previously, the surface represented by Eq. (3.10) may be integrated in finite sections. Based on the specified values of ρ and ζ, *Temperature* calculates an appropriate size for the surface sections to be integrated by *qrombPrism2*. An appropriate size is determined by calculating the frequency with which Eq. (3.10) oscillates in both the γ and β directions since this determines the ease with which

qrombPrism2 converges to a solution. The value of ρ determines the frequency in the γ direction, which approaches $\rho/2\pi$ as γ increases. Likewise, the frequency of oscillation in the β direction is determined by ζ and is exactly equal to $\zeta/2\pi$.

$$f(\rho,\zeta,\tau,\gamma,\beta) = \gamma K(\beta,\zeta)\frac{\bar{\bar{Q}}\left[1-e^{-\left(\gamma^2+\beta^2\right)\tau}\right]}{\gamma^2+\beta^2}J_0(\gamma\rho) \tag{3.10}$$

Temperature calculates these frequencies and, from this information, determines limits of integration over which the integrand only oscillates a few times. These limits are passed into *qrombPrism2*, which returns a converged solution for the section volume. *Temperature* also increments the limits of integration appropriately and monitors the section volumes to determine when to begin integrating along a new row (as described previously). Ultimately, *Temperature* determines when the solution to Eq. (2.69) is completely converged for the current value of ρ and ζ.

(4) The function, *qrombPrism2*, directs and monitors the process of integrating Eq. (3.10) between the finite limits of integration provided by *Temperature*. After being called by *Temperature*, it obtains an initial series of increasingly accurate volume approximations for the current surface section using *prism2*. The initial number of approximations in this series is determined by a variable, *K*, which has been assigned a value of 5 for all temperature calculations in this study. This set of approximations is passed to *polint* which uses a K^{th}-order polynomial curve fit routine to obtain a predicted volume based on this set of approximate volumes. The $(K+1)^{th}$ volume approximation is obtained from *prism2* and *polint* is, again, used to obtain an improved volume prediction. This process of obtaining a new

volume approximation and using *polint* to predict the actual volume based on the approximations is repeated until the percent difference between the most current prediction and the previous prediction is less than a specified value.

(5) The function, *prism2*, is called by *qrombPrism2*, which passes in the limits of integration for the current surface section as well as an integer that specifies which volume approximation to return. The first time *prism2* is called for a given set of integration limits, this integer is always equal to 1. This instructs *prism2* to return the first volume approximation, which is illustrated in Fig. 3.2. In each subsequent call to *prism2*, this integer is incremented by one, instructing *prism2* to divide each section into four new ones and calculate the next volume approximation. The value of S_n in Eq. (3.8) is updated by *prism2* with each iteration so that Eq. (3.10) does not need to be reevaluated for the same values of γ and β.

(6) The function, *InverseHankFourINTEGRAND*, receives values for ρ, ζ, γ, and β from *prism2* and calculates Eq. (3.10) for these values. The value of τ, also required to evaluate Eq. (3.10), is defined as a global constant, as mentioned previously. Using these values of ρ, ζ, γ, β, and τ, this function evaluates the integrand of Eq. (2.69) and returns the value to *prism2*. However, each call to *InverseHankFourINTEGRAND* requires evaluation of Eq. (2.70). Therefore, this function calls *DoubTranOfHG*, passing in the current values of γ and β needed to evaluate Eq. (2.70).

(7) Similar to the purpose of *Temperature*, which is to evaluate the surface integral in Eq. (2.69), the purpose of *DoubTranOfHG* is to evaluate the surface integral in

Eq. (2.70). This integral represents the forward Hankel and Fourier transforms of the dimensionless heat generation function. As discussed previously, a maximum dimensionless heat generation, QMAX, was defined to truncate the heat generation function in the immediate vicinity of the electrode perimeter since it approaches infinity at $\rho = 1$ and $\zeta = 0$. Despite such measures to make the evaluation of Eq. (2.70) more manageable, very high gradients in the heat generation still exist near $\rho = 1$ and $\zeta = 0$. Therefore, while the general procedure used by *DoubTranOfHG* to evaluate Eq. (2.70) is essentially identical to that of *Temperature*, a difference exists in the way *DoubTranOfHG* determines appropriate limits of integration and assesses convergence for the row of surface sections containing $\rho = 1$ and $\zeta = 0$. The values of γ and β, substituted into Eq. (2.70), determine the oscillation frequencies of the integrand, which are $\sim \gamma / 2\pi$ and $\beta / 2\pi$ in the ρ and ζ directions, respectively. Like *Temperature*, *DoubTranOfHG* calculates these oscillation frequencies for the integrand of Eq. (2.70), which is shown in Eq. (3.11), and uses these values to determine appropriate limits of integration.

$$f(\rho,\zeta,\gamma,\beta) = \rho J_0(\gamma\rho) K(\beta,\zeta) Q(\rho,\zeta) \tag{3.11}$$

However, instead of integrating over surface sections of the same size for the entire surface as *Temperature* does, *DoubTranOfHG* integrates over smaller surface sections along the row containing $\rho = 1$ and $\zeta = 0$ than it does along subsequent rows. The convergence criterion for the section volumes along this row must be proportionally smaller to avoid prematurely determining that the row volume is converged.

(8) The purpose of, *qrombPrism1*, is analogous to *qrombPrism2* in that it directs and monitors the process of integrating Eq. (3.11) between the finite limits of integration provided by *DoubTranOfHG*. Using volume approximations obtained from *prism1* for the current surface section as well as the polynomial curve fit routine, *polint*, *qrombPrism1* obtains an appropriately converged section volume that is returned to *DoubTranOfHG*.

(9) Similar to *prism2*, *prism1* follows the procedure illustrated in Fig. 3.2 to calculate approximate volumes for the current limits of integration. In addition to the integration limits, it receives an integer from *qrombPrism1* that specifies which volume approximation to return.

(10) The function, *HankFourINTEGRAND*, receives values for ρ, ζ, γ, and β from *prism1* and calculates Eq. (3.11) for these values. It then returns this value to *prism1*.

Evaluation of Eqs. (3.10) and (3.11) in steps (6) and (10), respectively, requires evaluation of the kernel for the Fourier transform, Eq. (2.22). A separate function was written that accepts values for β and ζ from either *InverseHankFourINTEGRAND* or *HankFourINTEGRAND* and evaluates Eq. (2.22) for these values. The value of the Biot number, also required for evaluation of the kernel, is specified as a global constant. Evaluation of Eqs. (3.10) and (3.11) also requires evaluation of the zero-order Bessel function of the first kind, J_0. This was accomplished using a function from *Numerical Recipes in C* called *bessj0*. See Appendix B for a full listing of the C program code.

3.5 Reduction of Computation Time

Since numerical evaluation of Eq. (2.69) for all values of ρ, ζ, and τ requires evaluation of the integrand and thus Eq. (2.70) for the same discrete set of γ and β values, calculation of more than one temperature for a given program execution results in repetitive calculations. This is a particularly important consideration since the evaluation of Eq. (2.70) is computationally demanding. An approach was, therefore, used in generating temperature profiles that greatly reduced computation time by avoiding the recalculation of previously calculated values. In describing this approach, it is helpful to point out that Eq. (2.22) can be written as a sum of two separate terms. Therefore, Eq. (2.70) can, likewise, be written as the sum of two terms as shown in Eq. (3.12).

$$\begin{aligned}\bar{\bar{Q}} = &\left(\frac{2\beta^2}{\pi\left(\beta^2+Bi^2\right)}\right)^{1/2}\int_0^\infty\int_0^\infty \rho J_0(\gamma\rho)\cos(\beta\zeta)Q(\rho,\zeta)\,d\rho d\zeta \\ &+\left(\frac{2Bi^2}{\pi\left(\beta^2+Bi^2\right)}\right)^{1/2}\int_0^\infty\int_0^\infty \rho J_0(\gamma\rho)\sin(\beta\zeta)Q(\rho,\zeta)\,d\rho d\zeta\end{aligned} \tag{3.12}$$

The above integrals, without their respective coefficients, were evaluated for several values of γ and β over the range of values needed to evaluate Eq. (2.69). These values were then stored in two separate data files. Storing the evaluated surface integrals in Eq. (3.12) without their coefficients allows these same data files to be used for all Biot numbers. Because the Biot number contains the convective heat transfer coefficient, h, evaluation of these integrals with their coefficients would require that a set of data files be generated for each value of h for which temperature profiles were generated. Thus, evaluating the integrals without their coefficients makes the data files general for any chosen conditions.

To generate these data files for the forward Hankel and Fourier transforms, the algorithm illustrated in Fig. 3.6 was modified so that *Load_array_2d* would directly call *DoubTranOfHG* instead of *Temperature*. In addition, *HankFourINTEGRAND* was modified so that, in place of evaluating Eq. (2.22), it would calculate either $\cos(\beta\zeta)$ or $\sin(\beta\zeta)$ depending on whether data was being generated for the first or second surface integral, respectively, in Eq. (3.12). Once the files were generated, the algorithm illustrated in Fig. 3.6 was, again, modified so that a function called *DTinterp* would be called instead of *DoubTranOfHG*. With the first call to *DTinterp*, this function reads values from each data file, multiplies the values by their respective coefficients shown in Eq. (3.12), and stores them in an array. In this and subsequent calls to *DTinterp*, the needed values are simply read from the array.

To allow for the possibility that evaluation of Eq. (2.69) may require Eq. (2.70) to be evaluated for some γ and β value not contained in the data files, a linear interpolation routine was written so that a reasonable estimate for this value could be obtained. As long as the distance between adjacent data points was small enough, a linear interpolation between these points would be sufficiently accurate to obtain the needed value. Thus, an appropriate step size was assessed qualitatively by generating surface plots of Eq. (2.70) for different step sizes and observing the smoothness of these plots. A suitable step size was considered to be one that is small enough to obtain a sufficiently smooth surface plot, but not so small that an unreasonable amount of time or an unreasonably large file was required to generate and store the data over the required range of γ and β values. The range of γ and β values necessary to obtain accurate results was determined by modifying the algorithm in Fig. 3.6 so that *Temperature* would write to a file the limits of

integration for the current surface section each time that it called *qrombPrism2*. In this way, it was possible to see, for any specified convergence criterion, typical maximum values for the limits of integration. For example, if in evaluating Eq. (2.69) it was decided that the volume along any given row was sufficiently converged when section volumes dropped below 0.001 (i.e. section volumes are negligible), the output file would provide typical values of γ and β beyond which surface section volumes are negligible. In this manner it was determined that sufficiently accurate dimensionless temperature values could be obtained if Eq. (2.69) were evaluated between the limits of integration $\gamma = \beta = 0$ and $\gamma = \beta = 100$. Based on this range of values, an appropriate step size for generating the data files described previously was determined to be 0.15625. Thus, both data files contain 641^2 (410 881) values. While this represents a large amount of computation time, this alternative approach for evaluating Eq. (2.69) saves an enormous amount of time when generating entire temperature profiles. As mentioned previously, this is because evaluation of Eq. (2.69) for different values of ρ, ζ, and τ requires evaluation of Eq. (2.70) for the same discrete set of γ and β values. Furthermore, since the initial investment of time to generate the data files only needs to be made once, a stricter convergence criterion for evaluating Eq. (2.70) may be used with only a small cost. Therefore, this approach yields much more accurate results in much less time.

CHAPTER 4 – TEMPERATURE PROFILES

4.1 Analytical Results

After developing an analytical solution to the bioheat equation and a suitable means of evaluating the solution as described in Chapters 2 and 3, respectively, temperature profiles were generated at three dimensionless times, τ = 0.3, 2.0, and 5.5, and for three convection coefficients, h = 1000, 2000, and 4000 W/m^{2}K. Property values were obtained from the literature [7-8] and are summarized in Table 4.1. Contour plots for all nine cases appear in Figs. 4.1 to 4.3 where Figs. 4.1a to 4.1c show profiles for h = 1000 W/m^{2}K, Figs. 4.2a to 4.2c for h = 2000 W/m^{2}K, and Figs. 4.3a to 4.3c for h = 4000 W/m^{2}K. Arranged in this manner, it is easy to observe the time progression of the temperature profile for any one of the three convection coefficients.

Table 4.1 Cardiac Tissue Properties [7-8].

Density, ρ_t	1100 kg/m^3
Specific Heat Capacity, c	3111 J/kgK
Thermal Conductivity, k	0.531 W/mK
Electrical Conductivity, σ	0.5 S/m

Above approximately 50°C or a dimensionless temperature of 0.35, permanent tissue damage is assumed to occur [2]. Therefore, the 50°C isotherm represents the assumed location of the lesion boundary. For convenience, the contours in Figs. 4.1 to 4.3 are shown in increments of 0.35 so that the outermost isotherm corresponds to the lesion boundary predicted by this model. Additionally, it is convenient to note that when using the properties in Table 4.1 and an electrode radius of 1.3 mm, the ablation time in

seconds is obtained by multiplying the dimensionless time, τ, by a conversion factor of 10.9 sec.

There are two trends in particular that one would expect to observe experimentally as the convection coefficient at the tissue surface increases. First, the lesion size for any given time will decrease as the convection coefficient increases. This can be observed in Figs. 4.1 to 4.3 for any one of the three ablation times for which profiles were generated. The trend is particularly easy to observe for $\tau = 5.5$. The second trend is related to the first and has to do with the size of the lesion near the tissue surface. Experimentally, one would expect the size of the lesion as viewed from the surface (at $\zeta = 0$) to become smaller as the convection coefficient increases. This was observed at each ablation time and is most obvious again for $\tau = 5.5$. In fact, at this time and for a convection coefficient of 4000 W/m^{2}K, the lesion diameter at $\zeta = 0$ is only slightly larger than that of the electrode. Recognizing that the edge of the electrode is located at $\rho = 1$ and $\zeta = 0$, this can be seen in Fig. 4.3c. On the other hand, for the case of $\tau = 5.5$ and $h = 1000$ W/m^{2}K, the lesion diameter at $\zeta = 0$ is about twice the electrode diameter. This is seen in Fig. 4.1c.

For $\tau = 0.3$, the highest temperature was observed to be located at about $\rho = 0.95$ and $\zeta = 0.05$ for a convection coefficient of 1000 W/m^{2}K. However, for $\tau = 5.5$ the highest temperature for the same convection coefficient occurs at $\rho = 0.05$ and about $\zeta = 0.5$. This behavior can be explained by the fact that for a flat-tipped electrode, the current density is highest along the perimeter of the electrode. This is known as the edge effect [5]. The heat generation is greatest at $\rho = 1$ and $\zeta = 0$ so that before significant heat transfer into the tissue and from the surface has occurred, the highest tissue temperatures

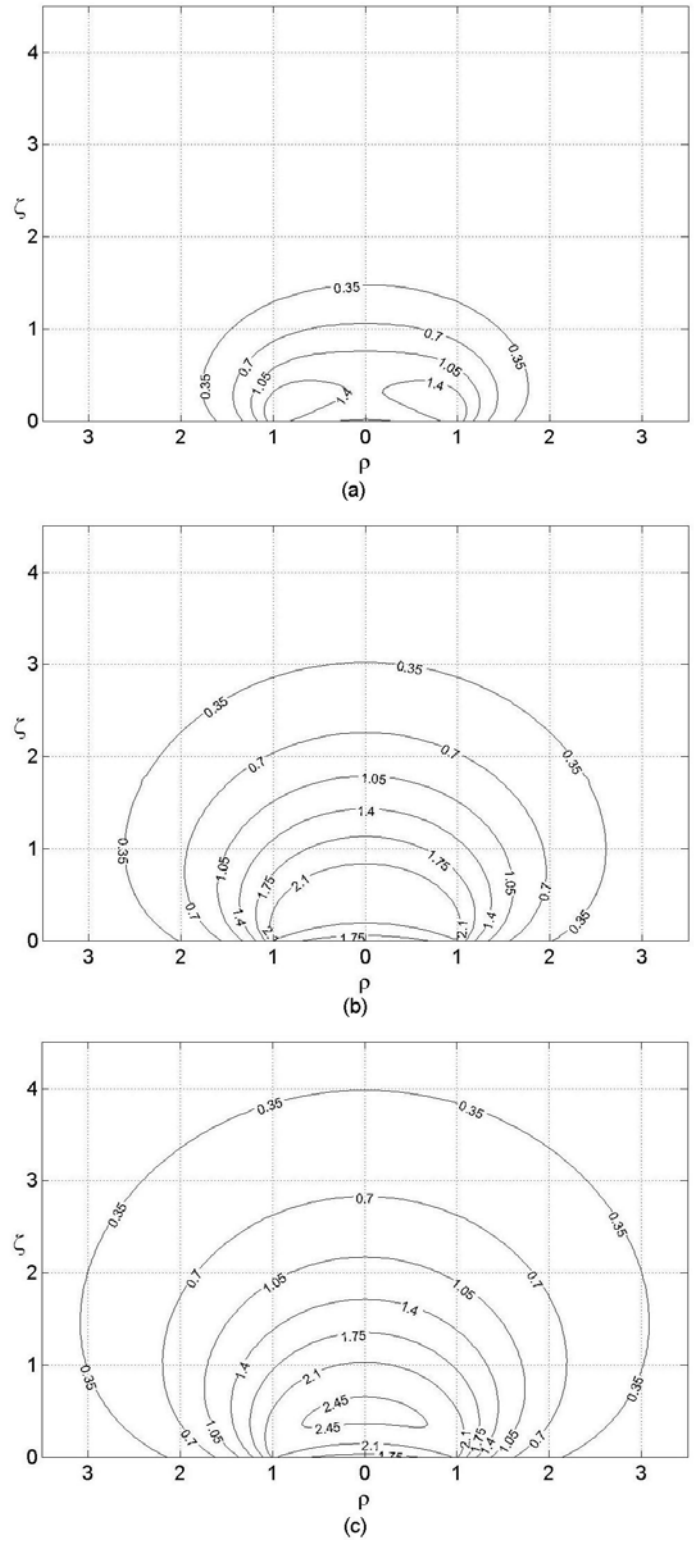

Figure 4.1 Dimensionless temperature profiles for h = 1000 W/m^{2}K and (a) τ = 0.3, (b) τ = 2.0, and (c) τ = 5.5.

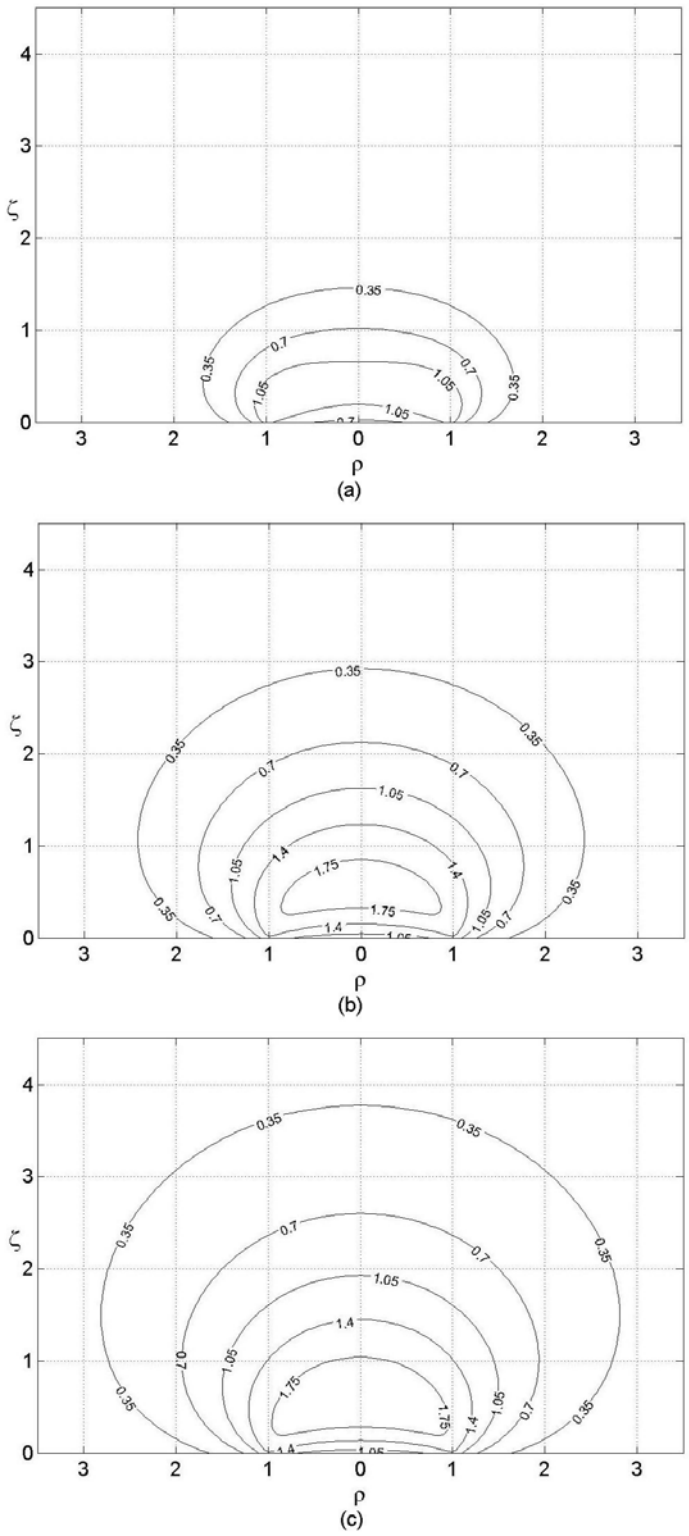

Figure 4.2 Dimensionless temperature profiles for h = 2000 W/m^{2}K and (a) τ = 0.3, (b) τ = 2.0, and (c) τ = 5.5.

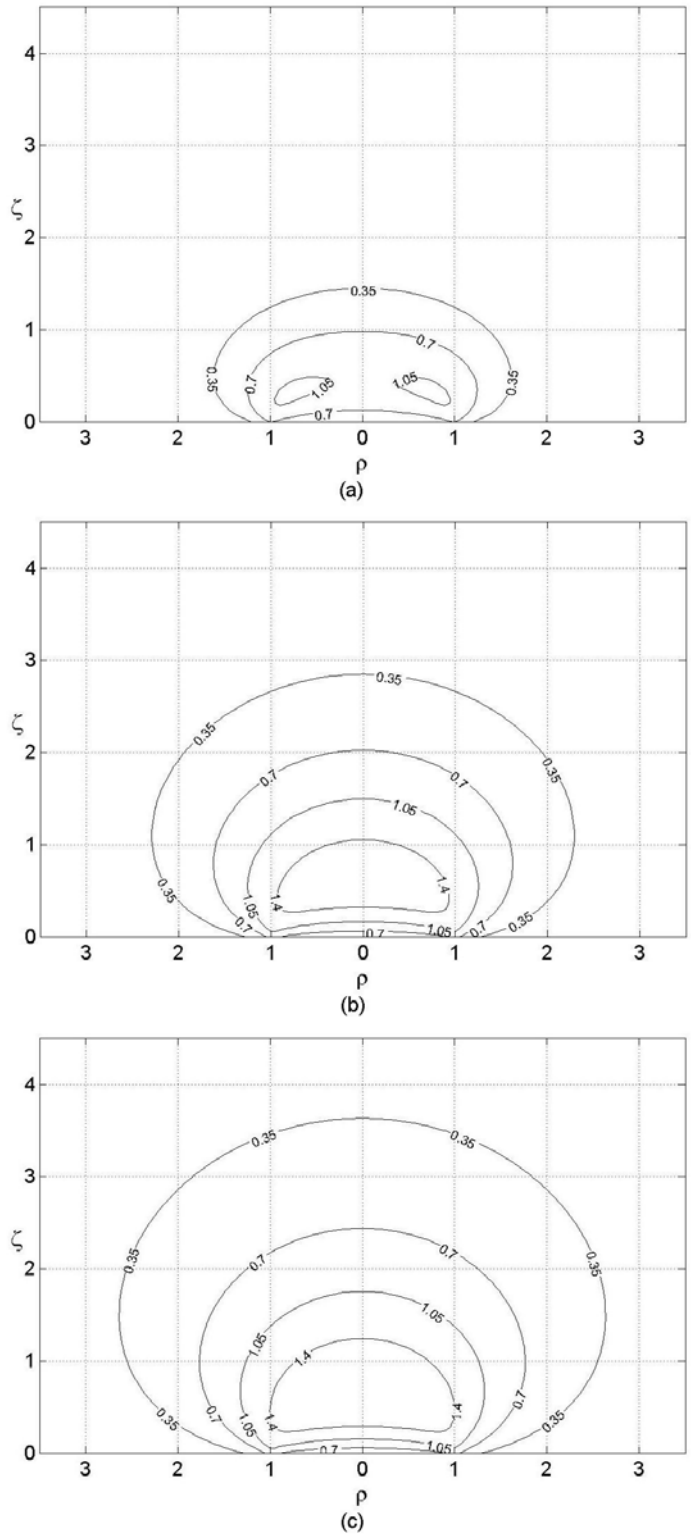

Figure 4.3 Dimensionless temperature profiles for h = 4000 W/m^{2}K and (a) τ = 0.3, (b) τ = 2.0, and (c) τ = 5.5.

occur near the perimeter of the electrode. As heat is conducted through the tissue and convected from the surface, the peak temperature moves toward $\rho = 0$ and further into the tissue. The instant ablation begins, the peak temperature occurs precisely at $\rho = 1$ and $\zeta = 0$.

Tungjitkusolmun et al. [7] provide a case to which the results of the current study can be compared. They simulate an ablation procedure in which the electrode is held at a constant rms voltage of 28 V for 60 seconds. For the current study, the electrode voltage, tissue properties, convection coefficient, and ablation time were chosen to be the same as those used by Tungjitkusolmun et al. [7]. The conditions for the case where τ is 5.5 (~60 sec) and the convection coefficient is 2000 W/m^2K are comparable. One significant difference between the two studies is that a flat-tipped electrode is assumed in the current study while Tungjitkusolmun et al. [7] assume a round-tipped electrode. Also, Tungjitkusolmun et al. [7] physically include the electrode in their model, thus accounting for conductive heat loss to the electrode.

The most significant differences in the results were in lesion width and maximum tissue temperature while the differences in lesion depth and location of maximum temperature were not so significant. Tungjitkusolmun et al. [7] reported values of 9.1 mm and 4.6 mm for the maximum lesion width and depth, respectively, while the current study found corresponding values of about 7.4 mm and 4.9 mm (differences of about -19 and +6.5 percent, respectively). It is believed these discrepancies are attributable to the difference in electrode shape. It seems that the regions in the tissue of highest current density when using a flat-tipped electrode would be relatively close to the axis, while the current density distribution for a round-tipped electrode would be more diffuse and not so

concentrated along the axis. This would explain why the lesion produced using a flat-tipped electrode is not as wide and extends further into the tissue.

The large discrepancy in maximum tissue temperature is likewise explained, in part, by the difference in electrode geometry. While Tungjitkusolmun et al. [7] report a value of 82.6°C, the maximum temperature observed in the current study was 110.9°C or about 28 degrees higher. As was mentioned, using a flat-tipped electrode instead of a round-tipped one results in a lesion that is 19 percent smaller in width. Assuming that the lesions are ellipsoid in shape, Eq. (4.1) gives their volume [17], where d is the lesion depth and w is the lesion width.

$$volume \approx \frac{4}{3}\pi d w^2 \tag{4.1}$$

Based on Eq. (4.1), it can be seen that a decrease of 19 percent in lesion width results in a lesion that is about 34 percent smaller in volume. This decrease in volume is partially offset by the fact that the lesion produced using a flat-tipped electrode is about 6.5 percent larger in depth. Accounting for this, the volume of the lesion produced using a flat-tipped electrode should be about 30 percent smaller than that produced using a round-tipped electrode. Thus, if the amount of energy entering the volume is comparable in both cases, a much greater average temperature increase would occur using a flat-tipped electrode.

The discrepancy in maximum tissue temperature is, also, attributable to the fact that the analytical model does not account for conduction of energy into the electrode, whereas Tungjitkusolmun et al. [7] do account for this. As will be seen in Section 5.5, the effect of conduction into the electrode is significant.

4.2 Numerical Results

To validate the numerical methods used to obtain the temperature profiles in Figs. 4.1 – 4.3, a numerical model was developed with a commercial software package (FLUENT®) using the same assumptions as were used in developing the analytical solution. Namely, constant tissue properties, negligible metabolic heat generation and heat loss due to blood perfusion, and a uniform convective boundary condition were assumed. The heat generation, Eq. (2.41), was incorporated into the model using a user-defined function (UDF) (See Appendix C). As seen in Fig. 4.4, the grids were constructed with a concentration of cells near the electrode. The cells are highly concentrated along the electrode perimeter. This was done in order to resolve the high gradients in heat generation near the electrode. The tissue section has two symmetry surfaces (sides), an adiabatic surface (bottom), and a surface with a convective boundary condition (top). The lengths of the straight edges in Fig. 4.4 are all 2 cm.

Three grids were created, one with 7290 cells, another with 26 264 cells, and the third with 58 320 cells. The coarsest grid was used in determining an appropriate time step for a 60 second simulation. In doing this, a convection coefficient of 2000 W/m^2K was used. Maximum tissue temperatures for cases where Δt = 0.5 sec and Δt = 1.0 sec were 382.68 K and 382.66 K, respectively. This represents a difference of 0.005%. Therefore, 1 second was assumed to be an appropriate time step. Simulations were then run using the three grids for the case where h = 2000 W/m^2K and t = 60 sec. Again, the maximum tissue temperatures were compared. These values, beginning with the coarsest grid were 382.7 K, 383.3 K, and 383.4 K. Based on the results of this grid refinement study, the finest of the three grids was assumed to be sufficient.

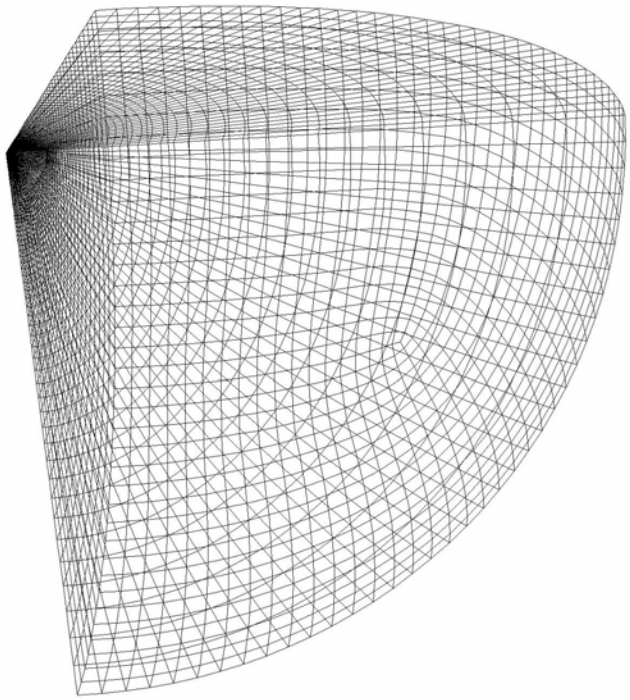

Figure 4.4 An image showing the general features of the mesh used for the numerical model. The tissue section has two symmetry surfaces (sides), an adiabatic surface (bottom), and a surface with a convective boundary condition (top). The lengths of the straight edges are all 2 cm.

Surface and axial temperature profiles predicted by the numerical model are shown in Fig. 4.5. The corresponding profiles predicted by the analytical solution are included as well for comparison. Excellent agreement in both surface (Fig. 4.5a) and axial (Fig. 4.5b) temperatures is observed except for a slight discrepancy in surface temperatures immediately near the electrode perimeter. This is most readily apparent in Fig. 4.6.

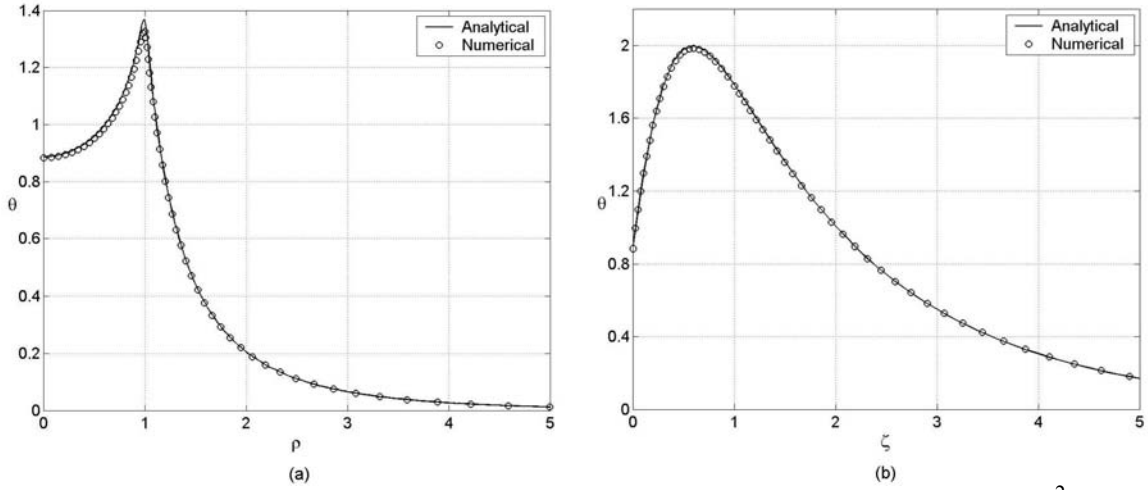

Figure 4.5 Dimensionless temperature profiles for $\tau = 5.5$ and $h = 2000$ W/m^{2}K (a) along the surface and (b) along the axis.

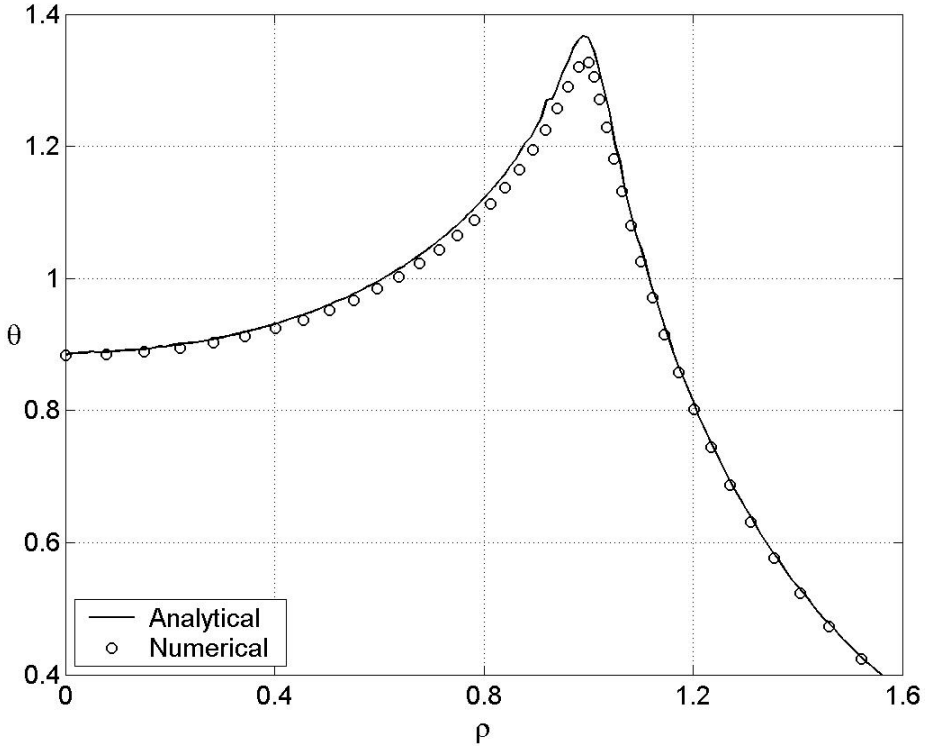

Figure 4.6 Dimensionless surface temperature profile for $\tau = 5.5$ and $h = 2000$ W/m^{2}K near the electrode edge.

Contour plots of both the numerical (Fig. 4.7a) and analytical (Fig. 4.7b) results for the case where τ = 5.5 and h = 2000 W/m^{2}K are shown in Fig. 4.7. There are no discernable differences between the two plots. Such close agreement between the analytical and numerical results validates the numerical methods used in evaluating the analytical solution to the bioheat equation.

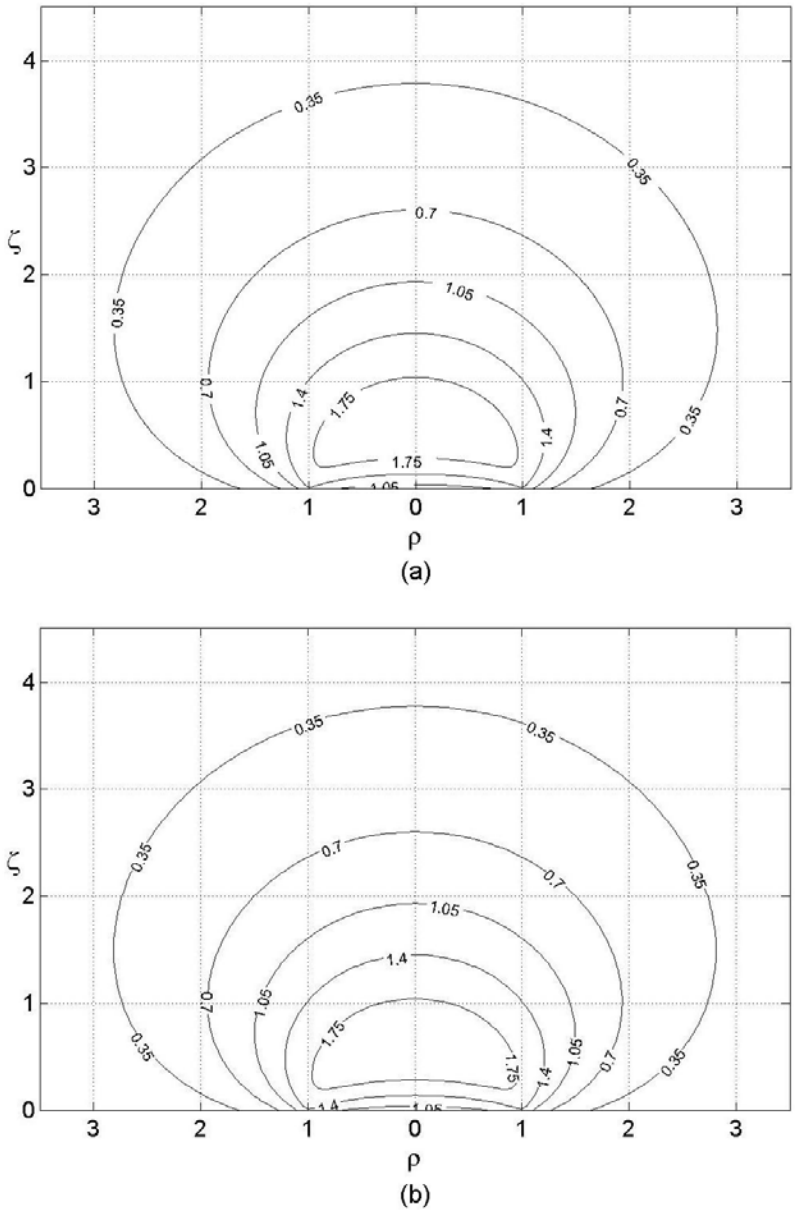

Figure 4.7 Dimensionless temperature profiles for τ = 5.5 and h = 2000 W/m^{2}K (a) predicted numerically and (b) predicted analytically.

The slight discrepancy between analytical and numerical modeling results near the electrode perimeter was assumed to be a result of not completely resolving the large gradients in heat generation in the numerical model. To investigate whether or not this was the case, a comparison was made between surface temperature profiles obtained using each of the three grids from the grid refinement study. Figure 4.8 shows the three profiles that were obtained numerically as well as the surface temperature profile obtained from the analytical solution. This figure shows that the numerical results approach the analytical results as the mesh size near the electrode is reduced.

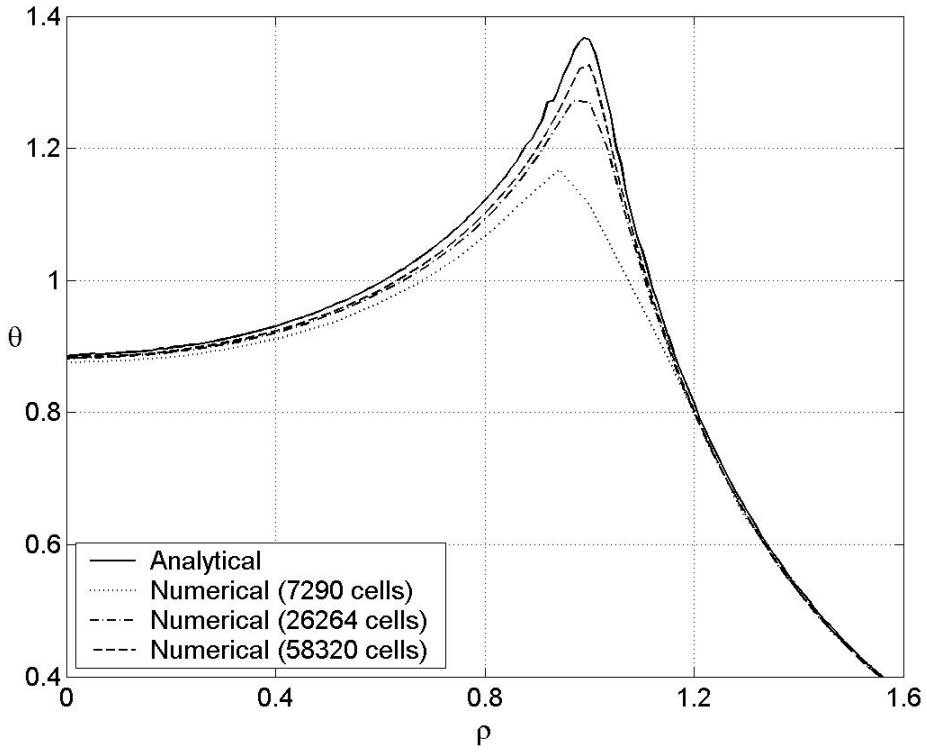

Figure 4.8 Plot of surface temperatures showing that the numerical results approach those of the analytical solution as the mesh is refined near the electrode.

Although an even finer grid near the electrode perimeter would further reduce the discrepancy, this seemed unnecessary based on results shown in Figs. 4.5 and 4.7.

Despite differences of up to about 1.5 degrees immediately near the perimeter, Fig. 4.5b shows that numerical temperature predictions along the axis are nearly identical to the analytical predictions. Furthermore, lesion dimensions predicted by the numerical model are nearly identical to the analytical predictions as seen by comparing Figs. 4.7a and 4.7b. This is also seen in Fig. 4.5. The ρ value in Fig. 4.5a and the ζ value in Fig. 4.5b at which dimensionless tissue temperatures fall below 0.35 (50^{o}C) represent the approximate locations of the lesion boundary on the tissue surface and along the axis, respectively. It can be seen from these plots that the ρ and ζ values predicted by the numerical and analytical solutions are nearly identical. This is significant since very accurate predictions of lesion geometry can be obtained numerically without completely resolving the high gradients in heat generation that occur using a flat-tipped electrode.

CHAPTER 5 – NUMERICAL VALIDATION OF MODEL ASSUMPTIONS

5.1 Discussion of Analytical Model Assumptions

This chapter addresses the third objective listed in Section 1.2 by investigating the validity of certain assumptions made in developing the analytical model described in Chapter 2. This investigation has been done using numerical modeling techniques. As mentioned in the introduction, the knowledge obtained from such numerical studies will be useful in assessing the quality of results obtained from an inverse heat transfer study. In addition, the following demonstrates how the analytical solution can serve as a benchmark for numerical studies to gain insights into RF ablation.

The approach for each numerical model was to relax only one or two of the assumptions made in the analytical model to isolate them and better assess the degree to which they influence results. Previous studies on RF ablation indicate which assumptions have the most significant effect on model accuracy. For example, the assumption that tissue properties are independent of temperature is good for all properties except for electrical conductivity [5,7]. A study by Tungjitkusolmun et al. [7] attempted to quantify changes in lesion geometry resulting from variations in electrical conductivity with temperature. Assuming a change in electrical conductivity of +2%/°C, Tungjitkusolmun et al. [7] showed a 57% increase in lesion volume and a 19% increase in maximum tissue temperature as compared with the case of constant electrical conductivity.

Another study by Jain and Wolf [4] assessed the effect of solving the conservation of mass, momentum, and energy equations for blood flow over the tissue surface instead of assuming a convective boundary condition. Their objective in doing this was to account for heating of the blood as it flows past the ablation site, which results in non-uniform conditions both upstream and downstream of the ablation site. Jain and Wolf show that, while some distortion of the lesion occurs, it is not significant for the blood velocities used. However, the authors do not present a clear basis for their selection of appropriate flow conditions. Section 5.3 describes a method for choosing flow conditions corresponding to a given convection coefficient that makes use of the analytical solution described in Chapters 2 and 3. This method is believed to be an improvement upon the work of Jain and Wolf.

Another important assumption of the analytical model is that the heat lost by convection from the portion of the tissue surface inside $\rho = 1$ is comparable to that which would be lost by conduction to the ablation electrode. This assumption is implicit in the fact that the electrode is not physically included in the analytical model. Rather, a uniform convective boundary condition is applied over the entire tissue surface. Previous numerical studies have physically included the electrode and have thus accounted for the conductive heat losses to the electrode.

5.2 Numerical Model Descriptions

Three numerical models have been developed to gain some insights into RF ablation and to address the most significant assumptions made in obtaining an analytical solution to the bioheat equation. These models have the following characteristics:

Model 1 This model assumes constant tissue properties, but solves the continuity, momentum, and energy equations for blood flow over the tissue surface instead of using a convective boundary condition. This accounts for blood heating, which results in non-uniform conditions at the surface. However, as with the analytical model, the electrode has been physically omitted. Therefore, this model does not account for the conduction of heat from the tissue into the electrode or effects associated with flow around the electrode.

Model 2 Like Model 1, this model assumes constant tissue properties and, also, solves the conservation equations for blood flow over the tissue surface. However, the electrode is physically included in this model to account for conduction of heat into the electrode and effects associated with flow around the electrode.

Model 3 Like the analytical model, this model neglects the presence of the ablation electrode and, instead, assumes a uniform convective boundary condition at the tissue surface. All tissue properties are assumed to be temperature-independent except for the electrical conductivity, which is assumed to increase by 2%/°C.

The results of these three models are summarized in Sections 5.4 to 5.6, which follow. Figure 5.2 (in Section 5.4) shows a schematic for Model 1 while Fig. 5.6 (in Section 5.5) shows one for Model 2. Note from these figures that the only difference between the two models is that Model 2 physically includes the ablation electrode while Model 1 does not. In numerically solving the conservation equations for the blood flow

in these two models, laminar flow is assumed. Also, it is important to recognize that, while blood heating results in different conditions upstream and downstream of the ablation site, a plane of symmetry exists that is parallel to the blood flow direction. Therefore, the region over which calculations were performed for Models 1 and 2 was divided in half, as depicted in Figs. 5.2 and 5.6.

For Model 3, a grid similar to that depicted in Fig. 4.4 was used. However, the boundary opposite the electrode was located 20 cm from the center of the electrode face for Model 3 instead of only 2 cm. For the numerical model described in Chapter 4, a distance of 2 cm was sufficiently far from the ablation site to be able to assign the temperature at this boundary to be body temperature, or 37°C. For Model 3, it was necessary to specify, not only thermal boundary conditions, but electrical boundary conditions, as well, to numerically calculate the energy generation in the tissue. The electric potential distribution decays much more slowly than the temperature distribution, thus, making it necessary to move the boundary opposite the electrode much further away so that conditions at this boundary would be very close to those that exist at infinity. As will be shown, assigning an electric potential of zero at this boundary, located 20 cm from the center of the electrode face, results in a heat generation profile nearly identical to that predicted analytically when assuming a constant electrical conductivity.

Because Models 1 and 2 make use of the heat generation function obtained analytically in Chapter 2, it was not necessary for the blood and tissue regions in these two models to be as large as the tissue region in Model 3. As shown in Figs. 5.2 and 5.6, the thickness and width of the blood and tissue for Models 1 and 2 are both 2 cm, while the length (in the streamwise direction) is 4 cm. The diameter of the electrode depicted

in Fig. 5.6 is 0.26 cm, which is small compared to the blood and tissue dimensions. For both Models 1 and 2, the heat generation in the tissue is calculated assuming this electrode diameter. It is important for the electrode diameter to be small compared to the blood and tissue dimensions since the models must closely approximate a semi-infinite region. In other words, all boundaries except for the inlet and outlet of the blood domain and the symmetry boundaries have to be far enough from the ablation site to be considered adiabatic. A negligible increase in temperature at these surfaces was observed at a steady-state condition when they were defined as adiabats. Therefore, the blood and tissue regions for Models 1 and 2 were assumed to be sufficiently large to model a semi-infinite region.

5.3 Determination of Appropriate Velocity Profiles

To properly model blood flow over the tissue surface, it was necessary to determine appropriate flow conditions corresponding to each of the convection coefficients used to generate the profiles shown in Chapter 4. The following describes a method for determining these flow conditions that is believed to be an improvement upon the approach used by Jain and Wolf [4]. As mentioned, Jain and Wolf do not present a clear basis for their selection of flow conditions. The approach used in the current study for determining such conditions is similar to experimental methods used to develop correlations that relate fluid velocity and properties to the convection coefficient. Such methods are based on the fact that Nusselt number is a function of both the Reynolds number and Prandtl number [18]. By varying the latter two parameters and measuring the resultant heat transfer rate from a surface, it is possible to develop expressions relating all three dimensionless parameters.

The current study used similar reasoning to obtain flow conditions corresponding to 1000, 2000, and 4000 W/m^2K by means of numerical experimentation. Because the Nusselt number indicates the magnitude of the heat flux at a location on the tissue surface, given the temperature difference between the surface and the free stream at that point, velocity profiles corresponding to these three convection coefficients were chosen by matching the total rate of heat transfer near the ablation site from numerical simulation to that obtained analytically. Characteristics of the velocity profiles were modified until the heat transfer rate obtained from numerical simulation was very close to that predicted by the analytical solution. For simplicity, heat transfer rates were obtained at a steady-state condition. Furthermore, they were only calculated for a small rectangular portion of the surface near the ablation site through which the heat flux is very high compared to that of the remaining surface (~50 times). The dimensions of this surface section are 1 cm in the streamwise direction and 0.5 cm in the cross-flow direction. The area of this section represents $1/16^{th}$ of the area of the entire blood-tissue interface.

5.3.1 Analytical Heat Rate Calculations

Equation (2.69) may be used to calculate the steady-state heat transfer rate through the small rectangular surface section described previously for any specified convective heat transfer coefficient. Allowing τ to approach infinity, the exponential term in Eq. (2.69) approaches zero so that the resulting steady-state temperature profile is described by Eq. (5.1).

$$\theta(\rho,\zeta) = \int_0^\infty \int_0^\infty \gamma K(\beta,\zeta) \frac{\bar{\bar{Q}}}{\gamma^2+\beta^2} J_0(\gamma\rho)\, d\beta\, d\gamma \tag{5.1}$$

Using the non-dimensional parameters defined in the nomenclature, Fourier's law can be non-dimensionalized to obtain Eq. (5.2), where Q'' represents a dimensionless heat flux.

$$Q''(\rho) = \frac{R}{kT_o} q''(r) = -\left.\frac{\partial \theta}{\partial \zeta}\right|_{\zeta=0} \tag{5.2}$$

This equation relates the heat flux through the blood-tissue interface to the dimensionless temperature gradient at the surface. Thus, an expression for heat flux through the tissue surface can be obtained by calculating the derivative of θ with respect to ζ and evaluating this expression for $\zeta = 0$. Doing so, and using Eq. (5.2), the following equation is obtained:

$$q''(r) = -\frac{kT_o}{R} \int_0^\infty \int_0^\infty \gamma \sqrt{\frac{2\beta^2 Bi^2}{\pi\left(\beta^2 + Bi^2\right)}} \frac{\bar{\bar{Q}}}{\gamma^2 + \beta^2} J_0\left(\gamma r / R\right) d\beta d\gamma \tag{5.3}$$

Equation (5.3) was evaluated for several values of r in the same manner that Eq. (2.69) was evaluated (described in Chapter 3). In other words, the heat flux was calculated at several discrete locations within the small surface section mentioned previously. In addition, the program used to evaluate Eq. (5.3) was modified so that each local heat flux value was multiplied by an appropriate area and summed to obtain the total steady-state heat transfer rate through the surface section. This was done for the three convection coefficients used previously (1000, 2000, and 4000 W/m^{2}K) and the results are summarized in Table 5.1.

Table 5.1 Steady-state heat transfer rates through a 1 cm by 0.5 cm surface section at the ablation site, calculated from the analytical solution. This was done for three convection coefficients: 1000, 2000, and 4000 W/m^{2}K.

Convection Coefficient (W/m^{2}K)	Heat Transfer Rate (W)
1000	0.681
2000	0.718
4000	0.739

The heat transfer rates summarized in Table 5.1 were used to determine appropriate velocity profiles corresponding to each of the three convection coefficients.

5.3.2 Numerical Heat Rate Calculations

Once heat transfer rates were obtained analytically, several steady-state numerical models were run, each with different flow conditions. In each of these models, a different free stream velocity, u_∞, and hydrodynamic boundary layer thickness, δ, were specified. The desired flow conditions were produced in FLUENT by specifying the inlet velocity profile using a UDF. For nodes located within the boundary layer, velocities were calculated using a sine function approximation for the profile. Otherwise, the UDF simply returned the free stream velocity. See Appendix C for a listing of the UDF code.

By varying both free stream velocity and boundary layer thickness, it was possible to obtain three sets of blood flow conditions for which the resulting steady-state heat transfer rates match the values shown in Table 5.1. It is important to point out that, for this process, the electrode was physically omitted as it is in Model 1. The flow conditions obtained are summarized in Table 5.2 with the corresponding convection coefficients shown in the first column. A plot containing the normalized velocity profiles is shown in Fig. 5.1. From this figure, the result of varying both the boundary layer thickness and the free stream velocity can be seen. For a relatively low convection coefficient of 1000 W/m^2K and a correspondingly low free stream velocity, the profile resembles a laminar profile. On the other hand, the profile corresponding to 4000 W/m^2K resembles a turbulent profile that is likely to exist for higher free stream velocities. Thus, the profile characteristics summarized in Table 5.2 and depicted in Fig. 5.1 are consistent with what is physically expected for different convection coefficients.

Table 5.2 Flow conditions corresponding to convective heat transfer coefficients of 1000, 2000, and 4000 W/m^{2}K.

Convection Coefficient, h (W/m^{2}K)	Boundary Layer Thickness, δ (cm)	Free Stream Velocity , u_∞ (m/s)
1000	2.0	0.21
2000	0.4	0.40
4000	0.1	1.30

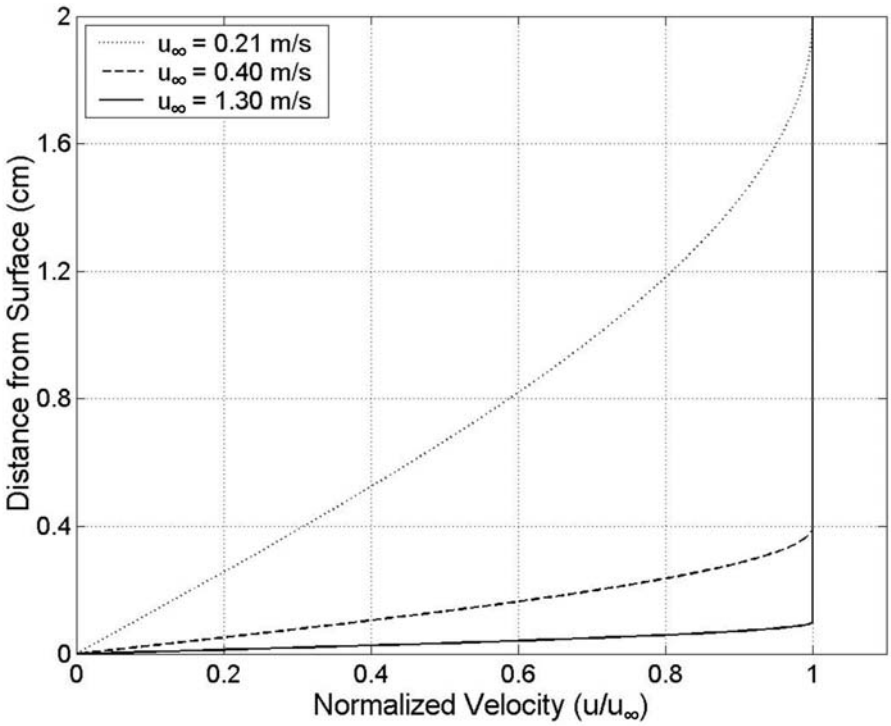

Figure 5.1 Normalized velocity profiles corresponding to three different convection coefficients. The free stream velocities (u_∞) for the profiles corresponding to 1000, 2000, and 4000 W/m^{2}K, are 0.21, 0.40, and 1.30 m/s, respectively.

The actual heat transfer rates obtained for each of these three sets of flow conditions are summarized in Table 5.3. Also included in Table 5.3 are the heat transfer rates obtained from the analytical solution and the percent difference between analytical and numerical values.

Table 5.3 Comparison of heat transfer rates obtained analytically and numerically for convection coefficients of 1000, 2000, and 4000 W/m^{2}K.

Convection Coefficient (W/m^{2}K)	Heat Rate (W) (Analytical)	Heat Rate (W) (Numerical)	Percent Difference
1000	0.681	0.681	0.029
2000	0.718	0.719	0.056
4000	0.739	0.738	0.149

Because the heat transfer rates calculated near the ablation site are so small, it was especially important to ensure that the values calculated numerically were obtained from a grid-independent solution. Therefore, a grid adaptation feature in FLUENT was used to produce four grids with progressively finer meshes in the vicinity of the electrode. The number of cells for each of these grids, in order of increasing cell number, is 61 246, 81 630, 175 024, and 536 350. Flow conditions were specified to be 1.6 cm and 0.2 m/s for the boundary layer thickness and the free stream velocity, respectively. Steady-state heat transfer rates through the small surface section were obtained for each of the grids. The results are summarized in Table 5.4. From this grid-refinement study, it was determined that the grid containing 175 024 cells was sufficient. The improved accuracy obtained by increasing the cell number from 175 024 to 536 350 did not justify the large increase in computation time. This grid was, therefore, used in obtaining the numerical heat transfer rates shown in Table 5.3 as well as the results for Model 1 summarized in Section 5.4.

Table 5.4 Results of a grid-refinement study that compares heat transfer rates near the ablation electrode. The boundary layer thickness, δ, and free stream velocity, u_∞, of the blood flow are 1.6 cm and 0.2 m/s, respectively.

Number of Cells	Heat Rate (W)	Percent Difference from Previous Value
61 246	0.652	–
81 630	0.675	3.58
175 024	0.685	1.47
536 350	0.690	0.73

5.4 Model 1 – Actual Flow with Electrode Omitted

Figure 5.2 is a schematic for Model 1 showing dimensions for the blood and tissue domains.

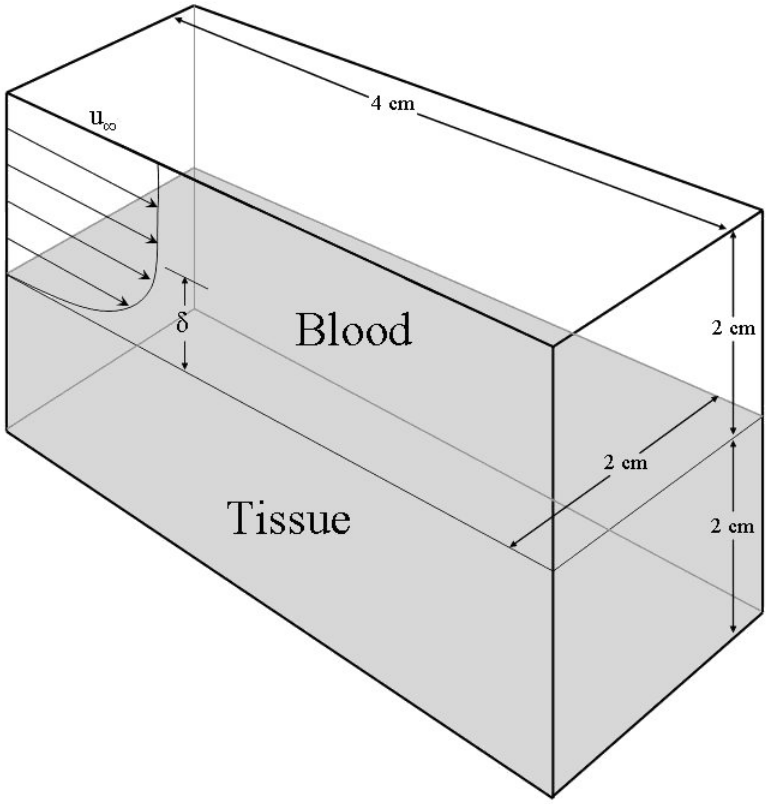

Figure 5.2 Schematic showing the main features of numerical model 1 with dimensions. The boundary layer thickness and free stream velocity of the blood are represented by δ and u_∞, respectively.

The purpose for obtaining flow conditions that resulted in heat transfer rates similar to those calculated analytically was so that the analytical temperature profiles shown in Chapter 4 could be used as a standard for comparison. It was expected that this process for choosing appropriate flow conditions would, for a given ablation time, result in lesions of comparable size to those shown in Figs. 4.1 – 4.3. Thus, the degree to which distortion of the lesions occurred could be better evaluated.

Temperature profiles were generated numerically for each of the three sets of flow conditions shown in Table 5.2 for ablation times of 3.2, 22, and 60 seconds. These times correspond to dimensionless times of 0.3, 2.0, and 5.5, which were used previously. Appropriate time steps for each of the three ablation times were determined using the flow conditions that correspond to a convection coefficient of 1000 W/m^2K. For each ablation time, three time step sizes were selected. These values are shown in Table 5.5.

Table 5.5 Time step sizes used in the temporal grid refinement study for Model 1. Heat transfer rates through the small surface section were compared using flow conditions for a convection coefficient of 1000 W/m^2K.

t = 3.2 sec	t = 22 sec	t = 60 sec
Δt = 0.1 sec	Δt = 0.5 sec	Δt = 1.0 sec
Δt = 0.2 sec	Δt = 1.0 sec	Δt = 2.0 sec
Δt = 0.4 sec	Δt = 2.0 sec	Δt = 5.0 sec

They are 0.1, 0.2, and 0.4 seconds for the shortest ablation time of 3.2 seconds, 0.5, 1.0, and 2.0 seconds for the ablation time of 22 seconds, and 1.0, 2.0, and 5.0 seconds for the ablation time of 60 seconds. For each ablation time, heat transfer rates through the small surface section were compared for the three time step sizes. These values are summarized in Table 5.6. From these results, it can be seen that the values obtained using the smallest time step are about 1.3%, 0.9%, and 0.6% greater than the values obtained using the largest time step for ablation times of 3.2, 22, and 60 seconds, respectively. The intermediate time step values were, therefore, considered appropriate.

Once appropriate flow conditions were determined that correspond to the convection coefficients used previously and spatial as well as temporal grid-refinement studies were performed, simulations were run for the nine cases discussed previously.

Table 5.6 Heat transfer rates through the small surface section for different step sizes. Flow conditions were used that correspond to a convection coefficient of 1000 W/m^2K.

t = 3.2 sec		
Δt = 0.1 sec 0.307 W	Δt = 0.2 sec 0.305 W	Δt = 0.4 sec 0.303 W
t = 22 sec		
Δt = 0.5 sec 0.548 W	Δt = 1.0 sec 0.546 W	Δt = 2.0 sec 0.543 W
t = 60 sec		
Δt = 1.0 sec 0.626 W	Δt = 2.0 sec 0.625 W	Δt = 5.0 sec 0.622 W

Figures 5.3 – 5.5 show results for these cases. Figure 5.3 contains profiles for the case where δ = 2 cm and u_∞ = 0.21 m/s, Fig. 5.4 for the case where δ = 0.40 cm and u_∞ = 0.40 m/s, and Fig. 5.5 for the case where δ = 0.10 cm and u_∞ = 1.30 m/s. For comparison, profiles obtained from the analytical solution for the corresponding set of conditions are, also, shown in the plots as gray contours. From these figures, it can be seen that the amount of lesion distortion decreases as blood velocity increases. The distortion is most obvious on the downstream side of the electrode, whereas on the upstream side, the numerical and analytical models match relatively well for all three sets of flow conditions. It is important to note that on the upstream side, the analytical model over-predicts temperatures, while on the downstream side, it under-predicts them. This can be explained by the fact that the thermal boundary layer on the upstream side is very small (a high temperature gradient exists) resulting in a higher rate of heat transfer than predicted by the analytical solution. On the other hand, the thermal boundary layer is larger (a lower temperature gradient exists) on the downstream side resulting in a lower rate of heat transfer than predicted by the analytical solution.

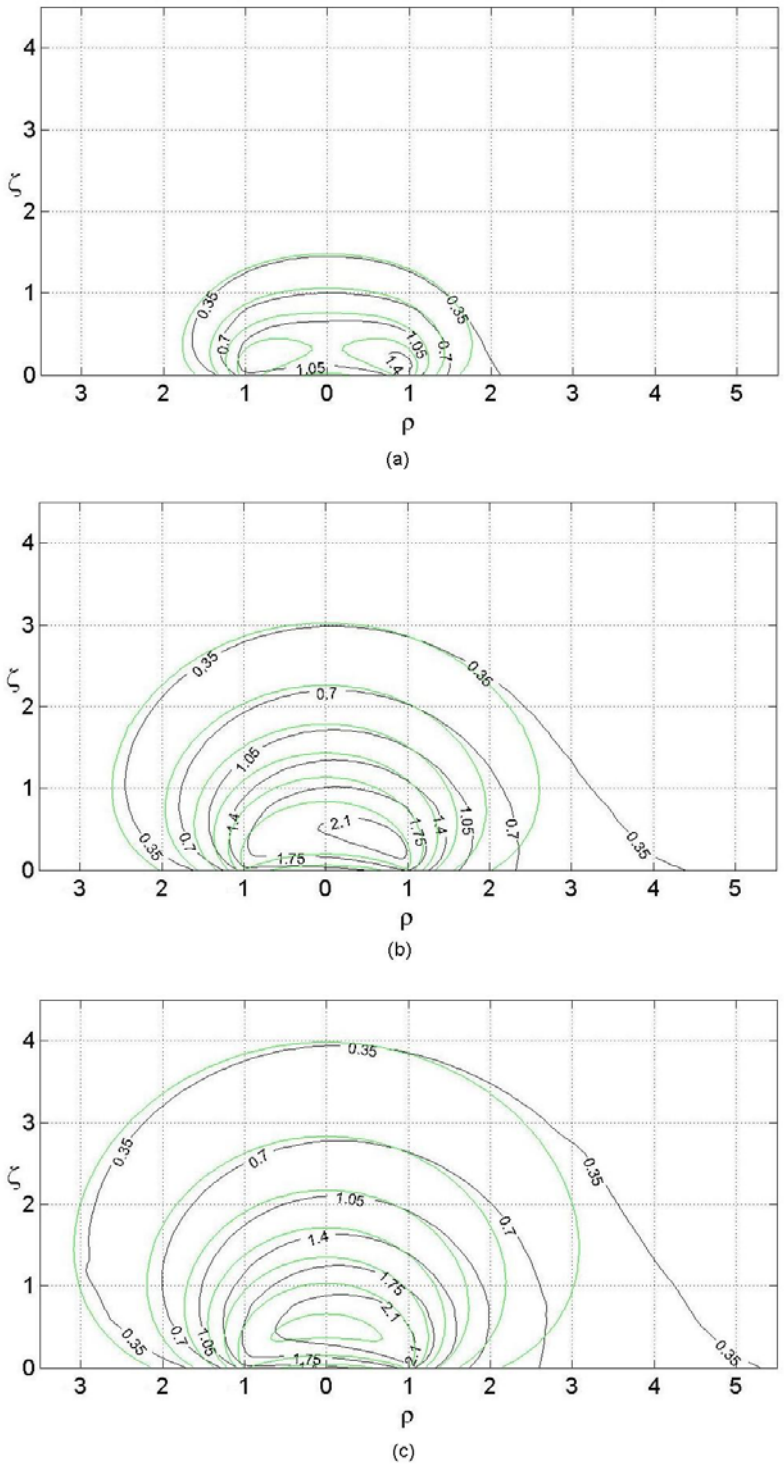

Figure 5.3 Temperature profiles obtained numerically for the case where $\delta = 2$ cm and $u_\infty = 0.21$ m/s for ablation times of (a) 3.2 sec, (b) 22 sec, and (c) 60 sec. Also shown, in gray contours, are temperature profiles obtained analytically for the corresponding convection coefficient of 1000 W/m^{2}K.

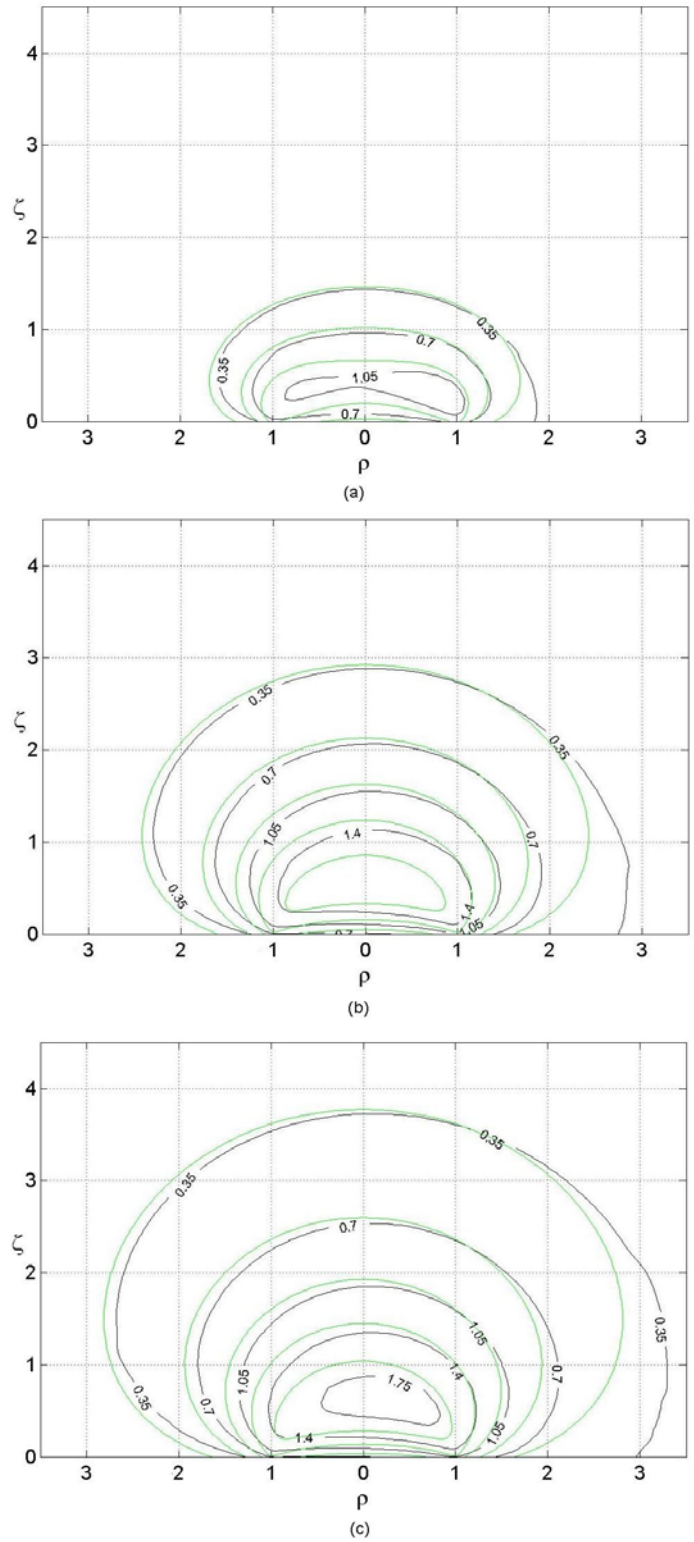

Figure 5.4 Temperature profiles obtained numerically for the case where δ = 0.40 cm and u_∞ = 0.40 m/s for ablation times of (a) 3.2 sec, (b) 22 sec, (c) and 60 sec. Also shown, in gray contours, are temperature profiles obtained analytically for the corresponding convection coefficient of 2000 W/m^{2}K.

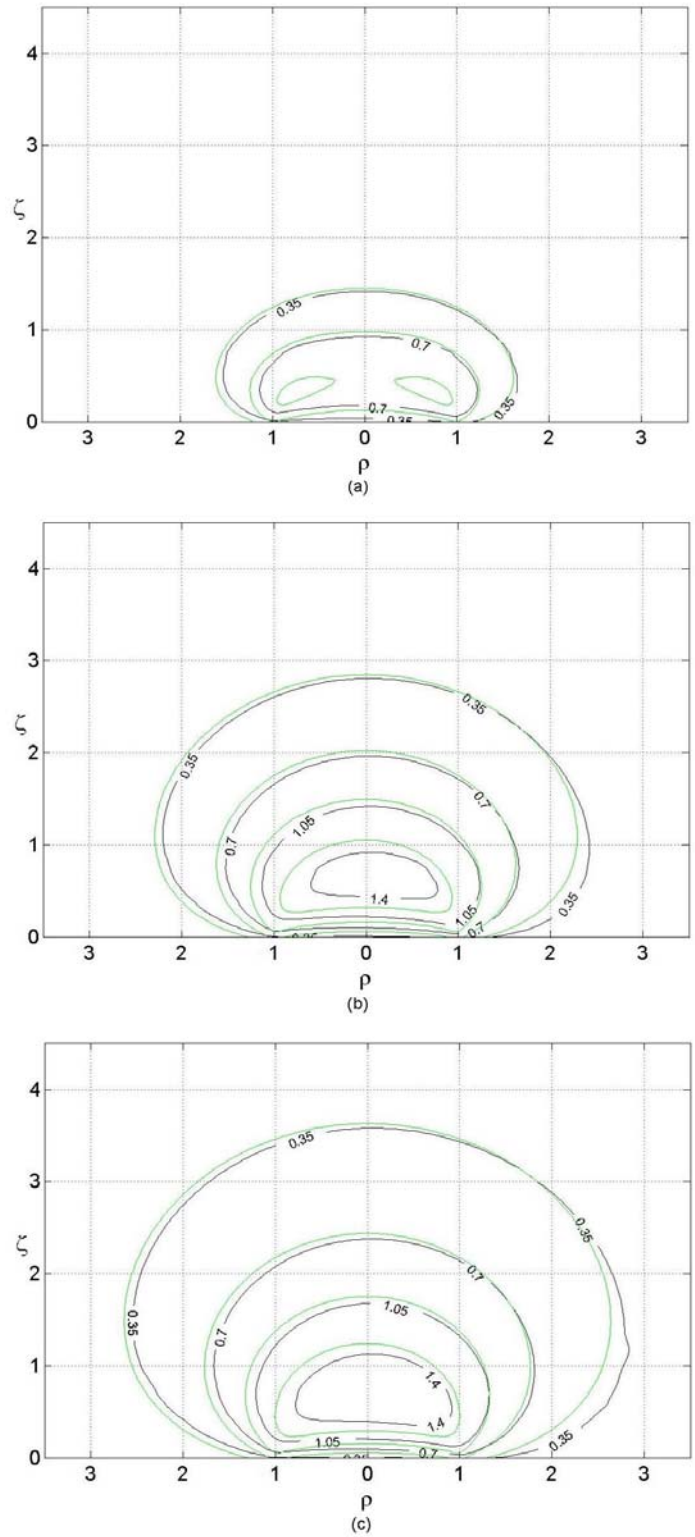

Figure 5.5 Temperature profiles obtained numerically for the case where δ = 0.10 cm and u_∞ = 1.30 m/s for ablation times of (a) 3.2 sec, (b) 22 sec, and (c) 60 sec. Also shown, in gray contours, are temperature profiles obtained analytically for the corresponding convection coefficient of 4000 W/m^{2}K.

The greater amount of lesion distortion that occurs for lower flow rates can be explained in a similar manner. The thickness of the thermal boundary layer depends on the time required for a given volume of blood to flow over the ablation site. At lower flow rates, thermal effects extend further into the flow, resulting in smaller temperature gradients and reduced convective heat transfer rates, particularly on the downstream side. This explains why the lesion distortion becomes more pronounced as the blood velocity is decreased. Comparing Figs. 5.3 and 5.5, a dramatic difference in lesion distortion is observed for flow conditions corresponding to 1000 W/m^2K and 4000 W/m^2K, respectively. Figure 5.3 shows that the analytical solution significantly under-predicts temperatures on the downstream side, resulting in less accurate predictions of lesion geometry. In contrast, Fig. 5.5 shows that very accurate predictions of lesion geometry can be obtained from the analytical solution for high convection coefficients.

5.5 Model 2 – Actual Flow with Electrode Included

Figure 5.6 is a schematic for Model 2 showing dimensions for the blood and tissue domains. Also, shown in this figure is the ablation electrode. The purpose of Model 2 was to investigate how lesion geometry is influenced by the ablation electrode. Model 1 did not consider the effects on lesion geometry resulting from conduction of heat into the electrode from the tissue or the significant changes in flow characteristics that occur near the ablation site. Model 2 accounts for both of these factors. A potentially important effect that is not considered in Model 2 is the resistive heating of the blood that occurs as it flows around the electrode.

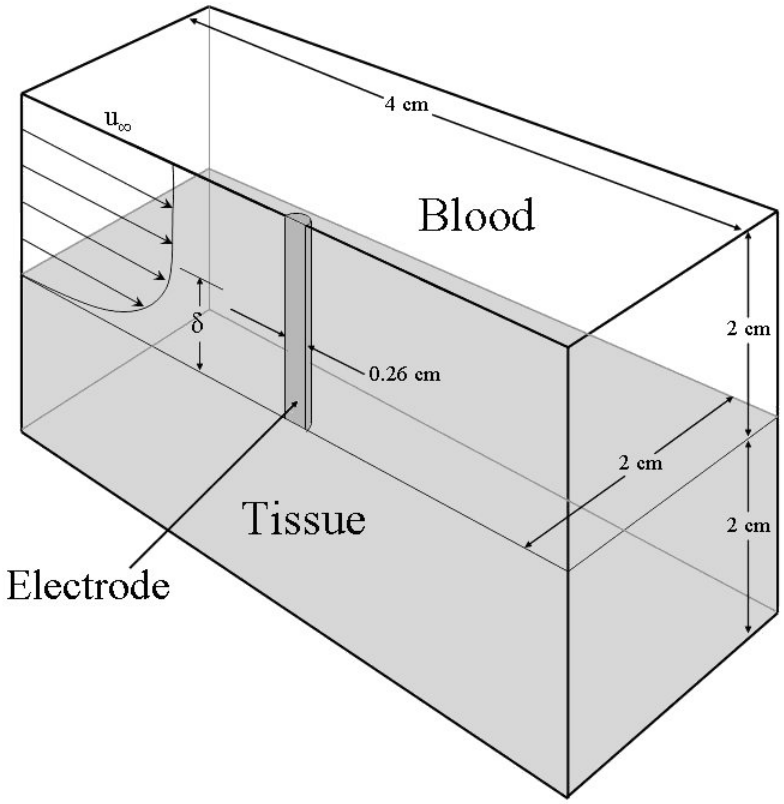

Figure 5.6 Schematic showing the main features of numerical model 2 with dimensions. The boundary layer thickness and free stream velocity of the blood are represented by δ and u_∞, respectively.

The properties of a Platinum-Iridium (Pt-Ir) ablation electrode are shown in Table 5.7 [5]. The thermally insulated core of the actual electrode, through which the thermistor wires run, was replaced by an air-filled core. This core is 0.5 mm in diameter [3].

Table 5.7 Ablation electrode properties (Pt-Ir) [5]

Density, ρ (kg/m^3)	Thermal Conductivity, k (W/m K)	Specific Heat, c (J/kg K)
21 500	73	131

The flow conditions shown in Table 5.2, which were used for Model 1, were again used for Model 2. The only difference between the two models was, therefore, the

physical presence of the electrode in Model 2. In addition, the time steps used for Model 1 were assumed to be appropriate for Model 2. Therefore, the only preliminary work that needed to be done before running Model 2 was a grid-refinement study. In this case, only two grids were used, one with 150 124 cells and the other with 336 681 cells. An ablation time of 60 seconds (Δt = 2 sec) and flow conditions corresponding to a convection coefficient of 2000 W/m^{2}K were used. Because the results were so similar using the two grids, it was not necessary to use more than two. Table 5.8 summarizes the results of the grid-refinement study, where maximum tissue temperature was used as a standard for comparison.

Table 5.8 Results of a grid-refinement study that compares maximum tissue temperature. An ablation time of 60 seconds (Δt = 2 sec) and flow conditions corresponding to a convection coefficient of 2000 W/m^{2}K were used.

Number of Cells	Maximum Tissue Temperature (°C)	Percent Difference from Previous Value
150 124	88.087	–
336 681	88.242	0.176

The difference in maximum temperature for the two grids is only 0.176%. Furthermore, Fig. 5.7 shows that there is only a slight difference in lesion geometry for the two cases. As a result, the grid containing 150 124 cells was considered appropriate.

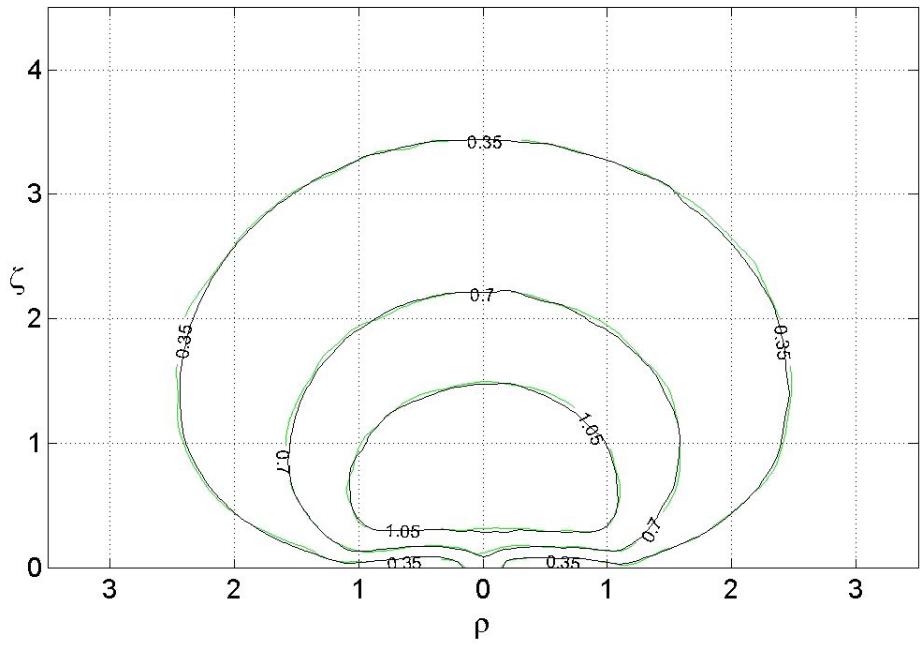

Figure 5.7 Temperature profiles for an ablation time of 60 seconds (Δt = 2 sec) using flow conditions that correspond to a convection coefficient of 2000 W/m^{2}K. The two profiles shown were obtained using grids of 150 124 and 336 681 cells.

Figure 5.8 shows results obtained for Model 2 for an ablation time of 60 seconds and convection coefficients of 1000, 2000, and 4000 W/m^{2}K. Both the numerical and corresponding analytical results are shown, where the gray contours are the analytical results. The results of Models 1 and 2 are also compared in Fig. 5.9, for an ablation time of 60 seconds and flow conditions corresponding to all three convection coefficients. Results from Model 1 in Fig. 5.9 are shown as gray contours.

An important observation that may be made, particularly in comparing the results of Models 1 and 2 shown in Fig. 5.9, is that the disruption in flow caused by physically including the electrode actually eliminates the distortion that resulted when the electrode was not included (shown in Figs. 5.4 – 5.6). In other words, the assumption that temperature profiles are axisymmetric (which was made in obtaining the analytical solution) is a good one.

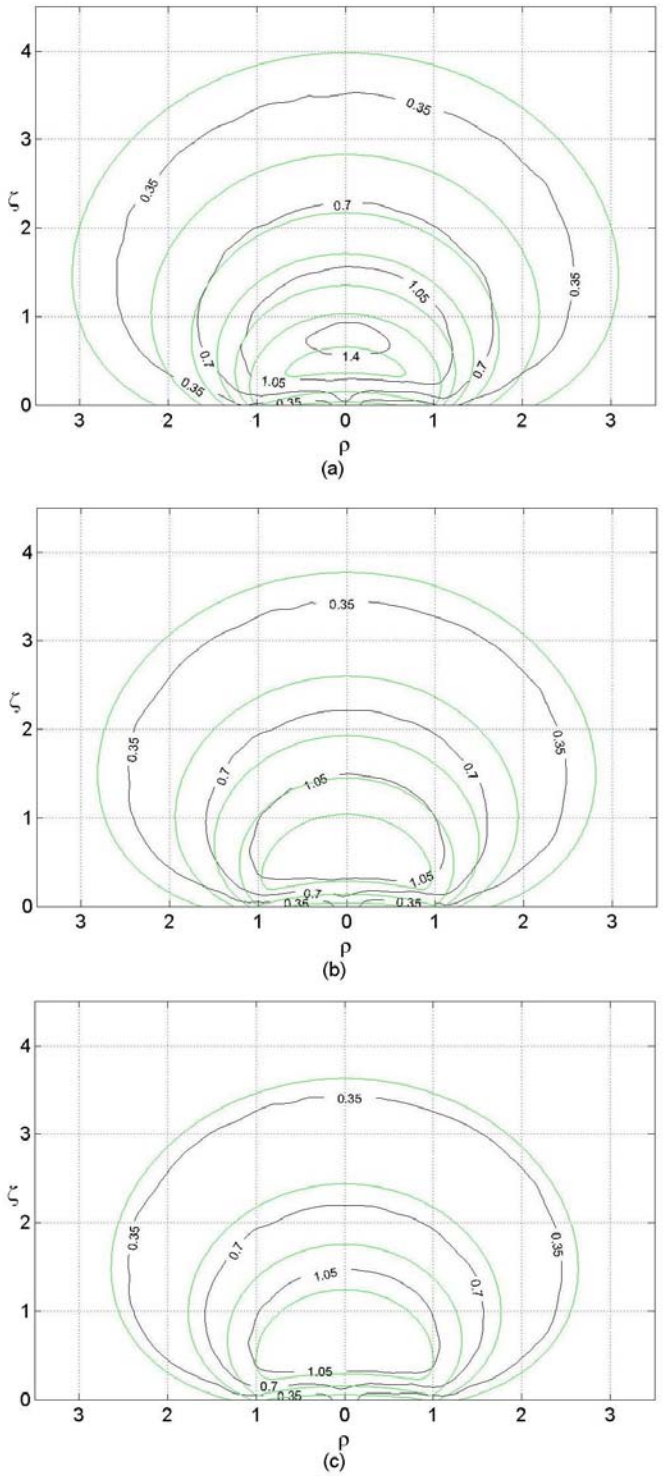

Figure 5.8 Results obtained for Model 2 for an ablation time of 60 seconds and convection coefficients of (a) 1000, (b) 2000, and (c) 4000 W/m^{2}K. Also included are the corresponding analytical results shown as gray contours.

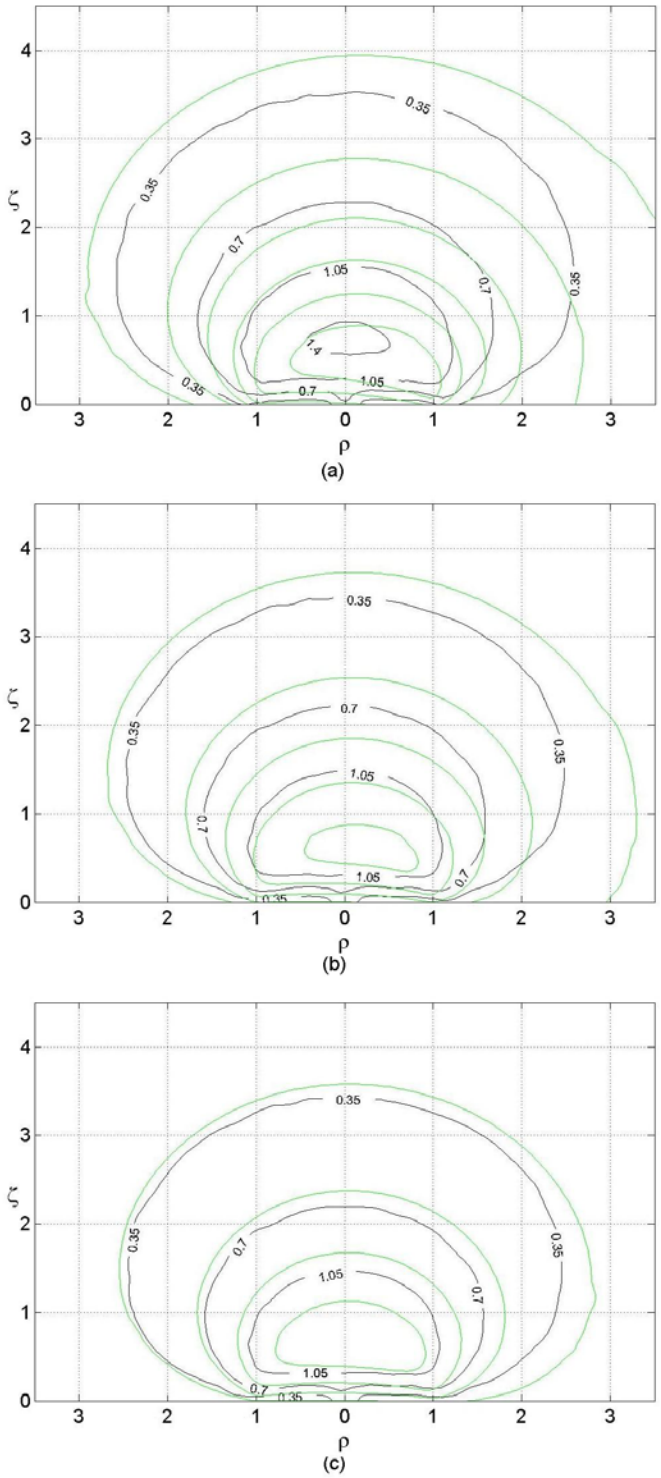

Figure 5.9 Results obtained for Model 2 for an ablation time of 60 seconds and convection coefficients of (a) 1000, (b) 2000, and (c) 4000 W/m^{2}K. Also included are the corresponding Model 1 results shown as gray contours.

This is consistent with results obtained by Jain and Wolf [4]. However, whereas the profiles in Figs. 5.8 and 5.9 show no discernible distortion, the results of Jain and Wolf show some distortion for all flow rates studied. This is explained by the fact that Model 2 does not account for resistive heating of the blood while Jain and Wolf do account for this. By comparison of the two models, resistive heating of the blood does affect lesion distortion, even if it is not very significant. The fact that some lesion distortion was observed by Jain and Wolf when accounting for resistive heating of the blood is consistent with what one would expect since stagnant flow conditions exist immediately upstream and downstream of the electrode.

Another observation that may be made from Figs. 5.8 and 5.9 is that accounting for conduction into the electrode results in much smaller lesions as compared to the analytical solution or Model 1. Therefore, conduction to the electrode is a dominant mode of heat transfer from the ablation site. The discrepancy in lesion size between Model 2 and the analytical solution could be reduced by accounting for resistive heating of the blood as Jain and Wolf did. However, while their results show that resistive heating of the blood affects lesion distortion, they do not address the issue of how resistive heating of the blood affects lesion size. When accounting for resistive heating, lesion sizes would be larger, but it is difficult to anticipate how much larger lesions would be as compared to the current case that does not account for resistive heating.

Another interesting result seen in Figs. 5.8 and 5.9 is that as the convection coefficient is increased, the lesion size predicted by Model 2 changes very little. The results of Model 2 show that the electrode temperature remains very close to the ambient body temperature of 37°C for convection coefficients over the range of 1000 to 4000

W/m^2K. Since conduction into the electrode is a dominant mode of heat transfer from the tissue surface, the minimal effect of the convection coefficient on electrode temperature explains why lesion size is not affected very much by the convection coefficient. Once again, however, it is important to consider the potential affects of resistive heating in the blood. The degree to which this heating occurs should be significantly influenced by the blood flow rate and, thus, the convection coefficient. Therefore, to gain an accurate understanding of how the convection coefficient influences lesion shape and size, it is necessary, as mentioned previously, to account for this effect.

5.6 Model 3 – Temperature-dependent Electrical Conductivity

Figure 5.10 is a schematic showing the main features of numerical model 3, including the thermal (Fig. 5.10a) and electrical (Fig. 5.10b) boundary conditions.

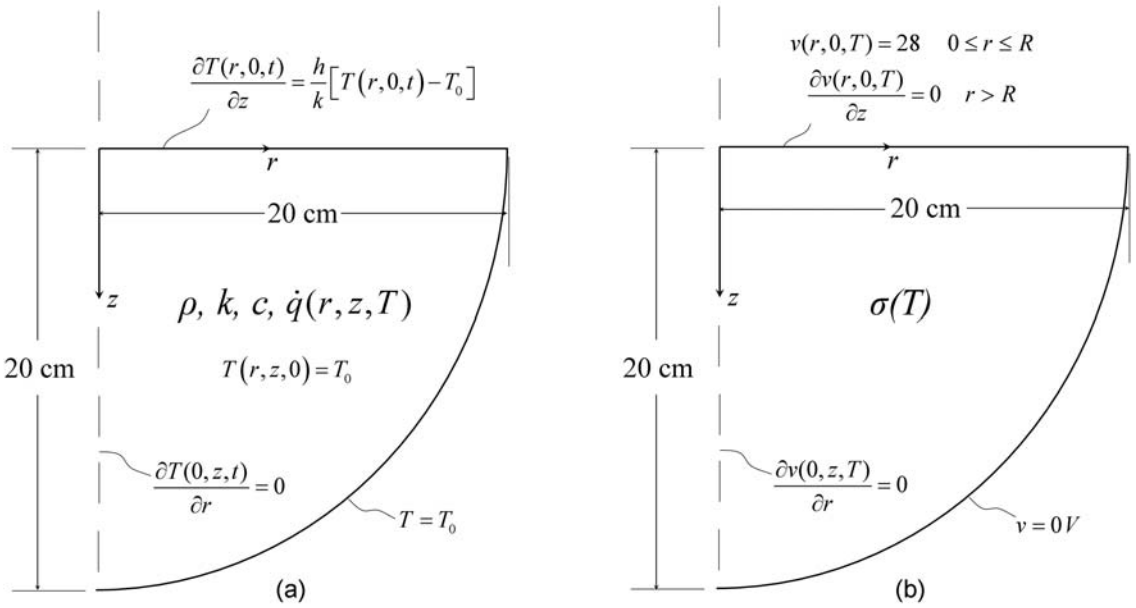

Figure 5.10 Schematic showing the main features of numerical model 3, including (a) thermal and (b) electrical boundary conditions.

Since Model 3 is similar to the numerical model described in Section 4.2, a grid similar to that shown in Fig. 4.4 was used for this model. However, as mentioned

previously, it was necessary to move the boundary opposite the electrode to a location 20 cm from the center of the electrode face. The reason for doing this was so that the potential at this boundary would more closely match the zero-potential condition that exists at infinity. The portion of the grid within 2 cm of the center of the electrode face was constructed in exactly the same manner as the finest grid for the model described in Chapter 4. Thus, it was not necessary to perform another grid-refinement study.

Because the purpose of Model 3 was to account for the dependence of electrical conductivity on temperature, the expression for heat generation developed previously (in Section 2.3) on the assumption of constant electrical conductivity could not be used. Solution of the Laplace equation to obtain the electric potential distribution had to be done numerically for each time step. This was accomplished by including the electric potential in the FLUENT model as a user-defined scalar (UDS). The equation for the electric potential could, thus, be solved simultaneously with the energy equation. A UDF was, once again, used to calculate heat generation rates for each cell using the local electric potential gradient (See Appendix C for a listing of the UDF code). As discussed in Chapter 2, the heat generation rate may be calculated from Eq. (2.11) at any location if both the electrical conductivity and the potential gradient are known at that location. Therefore, the local temperature (from which the electrical conductivity is calculated) and the potential gradient for each cell were passed to the UDF, which evaluated Eq. (2.11) using these values. Local heat generation rates were then returned to FLUENT. In this way, the coupled thermal-electric problem was solved.

The electrical conductivity was assumed to have a value of 0.5 S/m at 37°C and to increase by 2%/°C [7]. Therefore, the electrical conductivity as a function of temperature (in °C) is given by Eq. (5.4).

$$\sigma(T) = \sigma_{@37°C}\left[1 + 0.02(T - 37)\right] \tag{5.4}$$

To validate the approach just described for numerically solving the coupled thermal-electrical problem, the electrical conductivity was assigned a constant value of 0.5 S/m. The portion of tissue surface that the electrode would contact if it were included in the model was assigned a potential value of 28 V while the potential gradient over the remaining tissue surface was assigned a value of zero. As mentioned previously, the potential at the boundary opposite the ablation electrode was assigned a value of zero volts. As Figs. 5.11 and 5.12 show, the analytical and numerical results match very closely. Figure 5.11 shows axial heat generation rate profiles and Fig. 5.12 shows contour plots for an ablation time of 60 seconds and a convection coefficient of 2000 W/m^2K. In Fig. 5.12, the dashed contours were obtained numerically, while the solid contours were obtained analytically.

After validating the numerical methods used to obtain the heat generation in the tissue, temperature profiles were generated assuming the electrical conductivity varies with temperature according to Eq. (5.4). As mentioned previously, the electrical conductivity at 37°C was assumed to be 0.5 S/m. Temperature profiles were obtained for an ablation time of 60 seconds and convection coefficients of 1000, 2000, and 4000 W/m^2K. The results are shown in Figs. 5.13, 5.14, and 5.15, respectively. Figures 5.13a, 5.14a, and 5.15a show temperature distributions obtained analytically assuming a

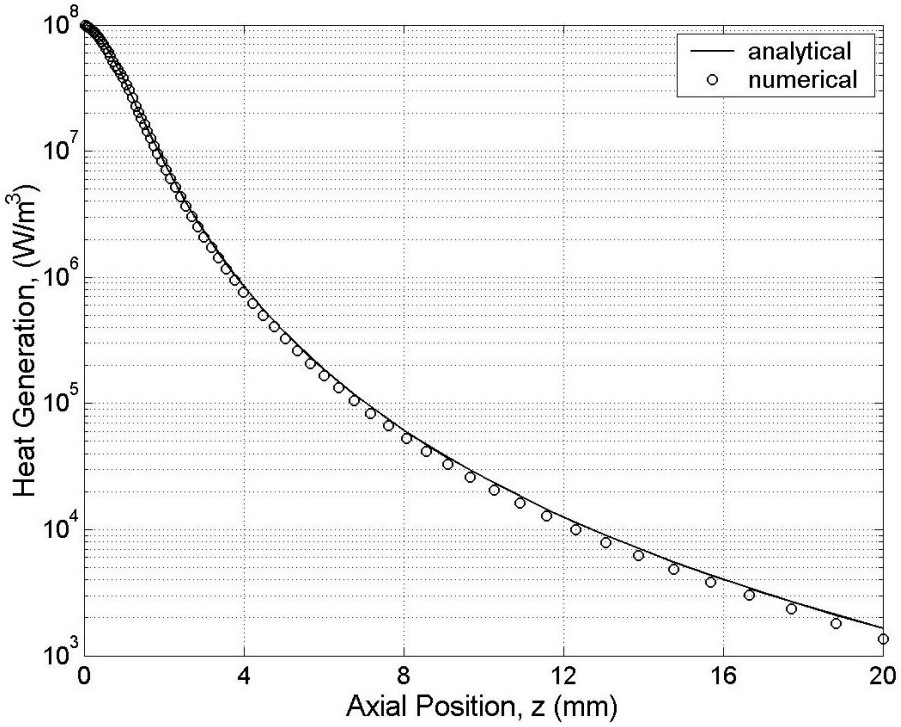

Figure 5.11 Heat generation rates predicted analytically and numerically for the case of a temperature-independent electrical conductivity.

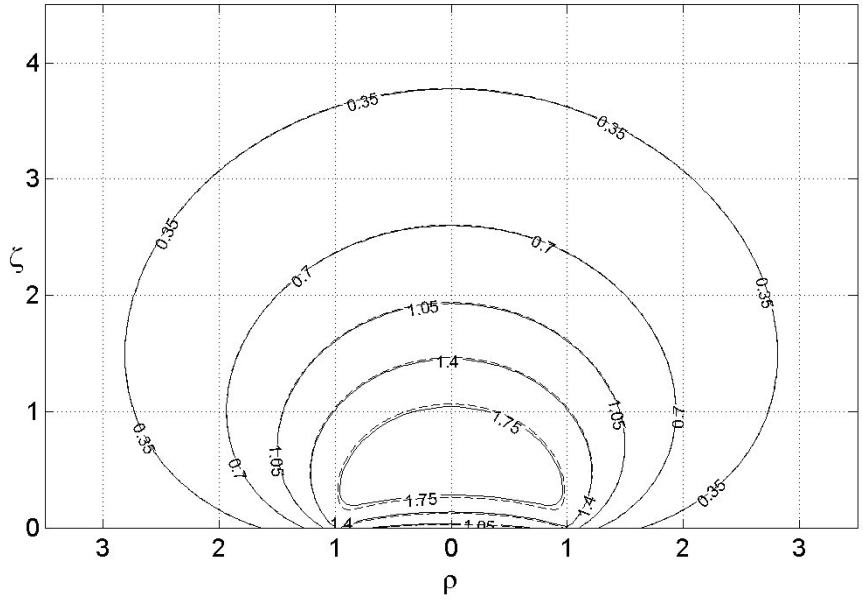

Figure 5.12 Temperature contours generated for a convection coefficient of 2000 W/m^2K and an ablation time of 60 seconds. The dashed contours were generated from a numerical model in which the heat generation rate was calculated numerically. The solid lines are contours obtained analytically.

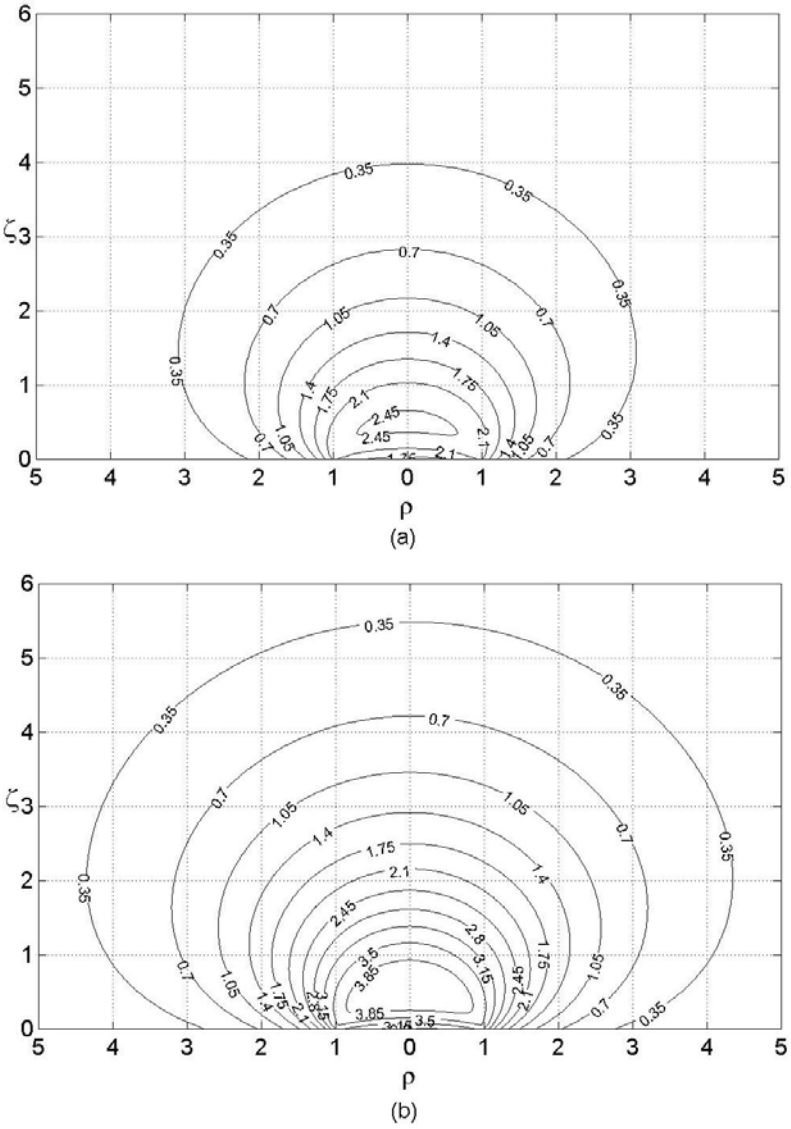

Figure 5.13 Contour plots for a convection coefficient of 1000 W/m^{2}K and an ablation time of 60 seconds assuming (a) a constant electrical conductivity and (b) an electrical conductivity that increases 2%/°C.

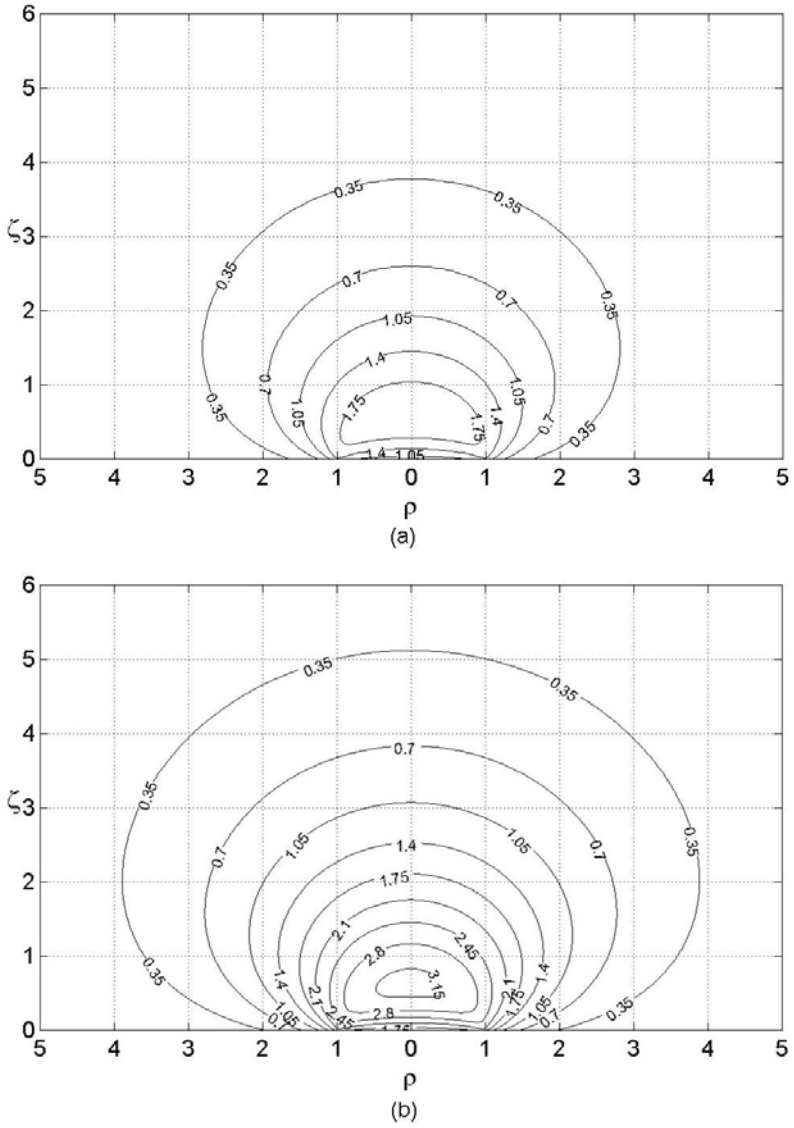

Figure 5.14 Contour plots for a convection coefficient of 2000 W/m^{2}K and an ablation time of 60 seconds assuming (a) a constant electrical conductivity and (b) an electrical conductivity that increases 2%/°C.

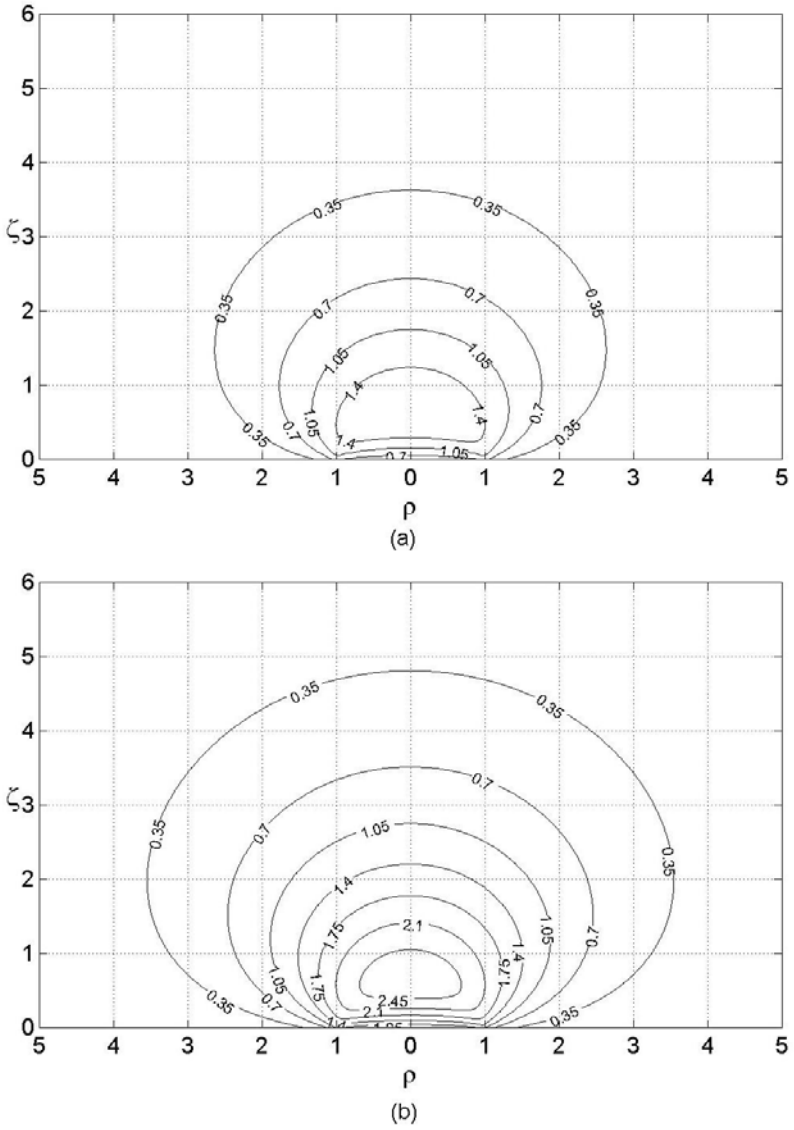

Figure 5.15 Contour plots for a convection coefficient of 4000 W/m^{2}K and an ablation time of 60 seconds assuming (a) a constant electrical conductivity and (b) an electrical conductivity that increases 2%/°C.

constant electrical conductivity and Figs. 5.13b, 5.14b, and 5.15b show the temperature distributions obtained numerically assuming a temperature-dependent electrical conductivity. Accounting for the temperature-dependence of electrical conductivity resulted in increases in maximum temperature of 62.4, 45.8, and 35.1°C for 1000, 2000, and 4000 W/m^2K, respectively. While it is important to note that the temperature distributions in Figs. 5.13b, 5.14b, and 5.15b could not represent reality (since charring and phase change of the water in the tissue would occur at such extreme temperatures), they do show qualitatively that lesion size and maximum temperature are dramatically affected by changes in electrical conductivity.

During an actual ablation procedure, excessively high tissue temperatures are avoided by the fact that the applied potential is continuously adjusted to maintain either a constant electrode tip temperature or a constant applied power. Furthermore, as the results to Model 2 show, conduction of heat into the ablation electrode (not accounted for in Model 3) significantly lowers tissue temperatures. Previous numerical studies that have accounted for these facts have reported much lower tissue temperatures [3-7]. For example, Tungjitkusolmun et al. [7] report an increase in maximum temperature of only 15.3°C for a power-controlled case in which the convection coefficient is assumed to be 2000 W/m^2K (electrical conductivity is assumed to increases 2%/°C).

As mentioned previously, the results in Figs. 5.13b – 5.15b were obtained assuming that the potential is zero at a distance of 20 cm from the center of the electrode face. According to Eq. (2.38), this potential is actually about 0.12 V. To assess the quality of this approximation, another grid was made where the boundary opposite the electrode was located 40 cm from the center of the electrode face instead of 20 cm. The

potential at this distance, as predicted by Eq. (2.38) is 0.056 V. A simulation was run for the case where the electrical conductivity is assumed to be independent of temperature. An ablation time of 60 seconds and a convection coefficient of 2000 W/m^{2}K were used. Upon comparison of these results with those obtained for the same conditions, but with the boundary located 20 cm from the center of the electrode face, a decrease of only 0.2°C in the maximum temperature was observed. Thus, a distance of 20 cm was assumed to be sufficiently large.

An interesting observation is made by comparing the electric potential and heat generation profiles for the constant-conductivity case to those for the variable-conductivity case. These profiles are shown in Figs. 5.16 and 5.17, respectively.

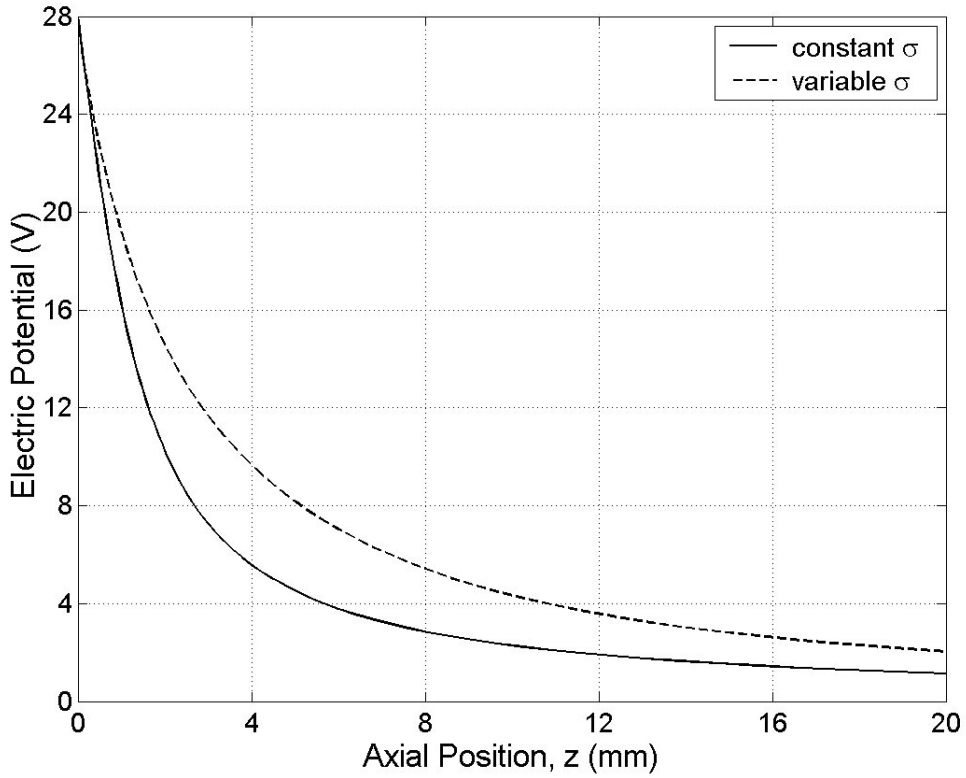

Figure 5.16 Electric potential distribution in the tissue along the axis ($\rho = 0$) after 60 seconds of ablation using a convection coefficient of 2000 W/m^{2}K. The dashed line shows the distribution assuming a temperature dependence of +2%/°C for the electrical conductivity and the solid line for a constant conductivity.

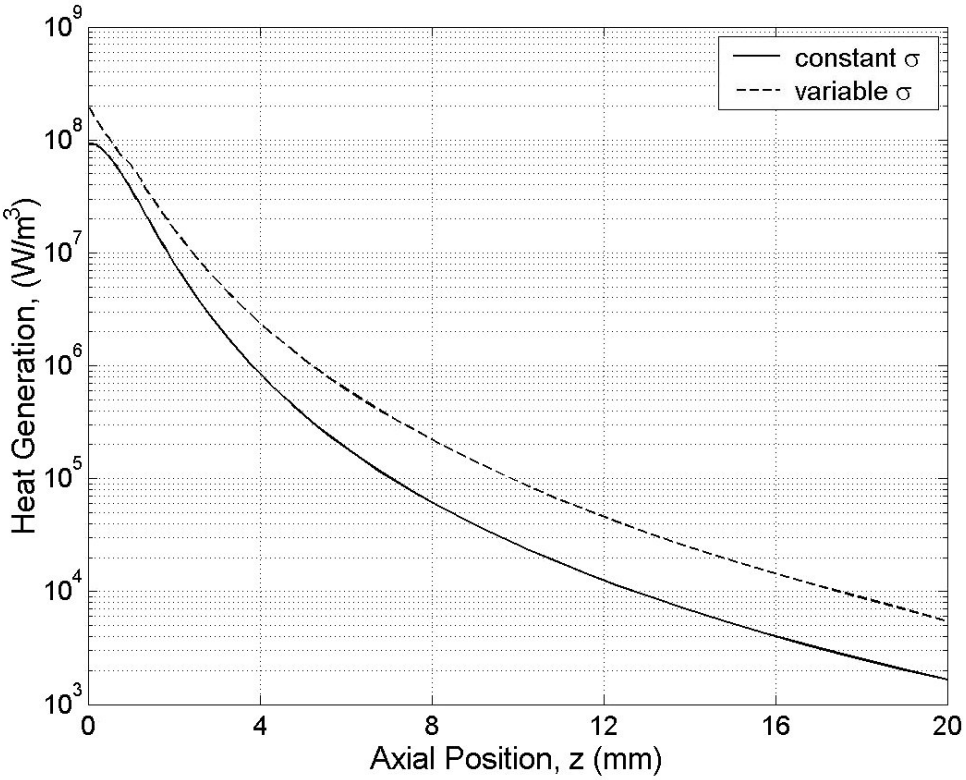

Figure 5.17 Heat generation rate in the tissue along the axis ($\rho = 0$) after 60 seconds of ablation using a convection coefficient of 2000 W/m^{2}K. The dashed line shows the distribution assuming a temperature dependence of +2%/°C for the electrical conductivity and the solid line for a constant conductivity.

Based on the fact that the slope in the electric potential near the electrode is greater in magnitude for the constant-conductivity case, it seems the heat generation would, likewise, be greater. This is because, according to Eq. (2.11), the heat generation is proportional to the square of the potential gradient. However, the degree to which the electrical conductivity increases with temperature appears to offset this fact. The result is that the heat generation is higher along the entire axis for the case where electrical conductivity is temperature-dependent. This can be seen in Fig. 5.17.

CHAPTER 6 – CONCLUSIONS AND RECOMMENDATIONS

An analytical solution to the bioheat equation has been developed using integral transforms. Important assumptions that were made in obtaining this solution include the following:

(1) Tissue properties are uniform and do not change with time

(2) The tissue temperature distribution is axisymmetric

(3) Metabolic heat generation and heat loss due to blood perfusion are negligible

(4) Heat loss at the tissue surface can be accurately modeled using a convective boundary condition with a uniform convection coefficient

(5) The heat loss that would occur to the ablation electrode at $\zeta = 0$ and $\rho < 1$ is comparable to that which occurs when a convective boundary condition is assumed within this region.

(6) The heat generation is accurately modeled by assuming the electrode is a flat disk and that the voltage field is equivalent to the steady field produced when the potential at the electrode is held at a constant rms voltage.

Once an analytical solution was obtained, an integration routine was written in C that is capable of evaluating surface integrals. This routine was validated using a surface integral with a known analytical solution and several temperature profiles were generated for various times and convection coefficients. To reduce computation time while improving the accuracy of temperatures calculated from the analytical solution, Eq.

(2.70) was evaluated for several γ and β values and these values were stored in data files. In this way it was possible to avoid having to recalculate them several times. Upon execution of the program, the data files were read and their contents stored in arrays for use in generating temperature profiles. Upon comparison of this temperature data to that obtained for a similar case using a round-tipped electrode, significant differences were found particularly for the lesion width and maximum temperature. These discrepancies are explained, in part, by the difference in electrode shape. Numerical modeling results, also, show that the discrepancies are largely explained by the fact that the analytical model does not account for conduction of heat into the electrode.

A numerical model was developed using the same assumptions used to obtain the analytical solution. This was done to validate the numerical methods used to evaluate the analytical solution of the bioheat equation. Agreement between the analytical and numerical solutions is excellent except for a slight discrepancy in surface temperatures in the immediate vicinity of the ablation electrode edge. This is explained by the fact that the mesh used in the numerical model does not completely resolve the high gradients in heat generation that exist near the electrode perimeter. Nevertheless, there is excellent agreement in lesion dimensions as predicted by the two models.

Three more numerical models were developed to assess the validity of some of the assumptions made in obtaining the analytical solution and to gain insight into RF ablation. The three models have the following characteristics:

Model 1 Assumes constant tissue properties, but replaces the convective boundary condition at the tissue surface with blood flow to account for blood heating. The electrode is physically omitted.

Model 2 Identical to Model 1 except that the electrode is physically included to account for conduction of energy into the electrode and effects from flow disruption.

Model 3 All tissue properties are assumed to be temperature-independent except for electrical conductivity. The electrode is physically omitted and a uniform convective boundary condition is assumed at the tissue surface.

Appropriate flow conditions used for Models 1 and 2 were determined using an approach similar to experimental methods used to develop correlations relating fluid velocity and properties to the convection coefficient. This approach is considered an improvement upon that used by Jain and Wolf [4]. While the results of Model 1 indicate that the assumption of uniform conditions at the tissue surface (and, thus, that tissue temperature distributions are axisymmetric) is only good for convection coefficients greater than about 4000 W/m^2K, inclusion of the electrode results in very negligible lesion distortion for convection coefficients between 1000 and 4000 W/m^2K. Thus, the assumption of axisymmetric temperature profiles is a good one. This is consistent with results reported by Jain and Wolf.

Comparing the results of Model 2 with those of Model 1 and the analytical solution, a noticeable difference in lesion size is observed. Particularly for convection coefficients of 1000 and 2000 W/m^2K, lesions as predicted by Model 2 are much smaller. Thus, conduction of heat into the electrode has a significant impact on lesion size. A potentially important effect that was not considered, however, in either of these two models is the resistive heating of the blood that occurs near the electrode. This is, likely, significant since there are regions of relatively stagnant blood immediately upstream and

downstream of the electrode. The effect of resistive heating in the blood would be to reduce the rate of heat transfer from the tissue surface and, thus, increase tissue temperatures. This, in turn, would reduce the discrepancy between analytical and numerical modeling results. Therefore, to better assess the feasibility of using the analytical solution in solving an inverse heat transfer problem, it is recommended that further numerical modeling be done to determine how significant this effect is.

The results of Model 3 indicate that lesion size and maximum temperature are strongly influenced by the temperature-dependence of electrical conductivity. Assuming electrical conductivity increases 2%/°C, significantly higher values for lesion size and maximum temperature are observed. Accounting for the temperature-dependence of electrical conductivity resulted in increases in maximum temperature of 62.4, 45.8, and 35.1°C for 1000, 2000, and 4000 W/m^2K, respectively. While the observed temperatures would not, in practice, be possible because of charring, phase change of water in the tissue, and monitors that control the electrode potential, the results of Model 3 qualitatively indicate the degree to which temperature distributions are influenced by the temperature-dependence of electrical conductivity. These results are consistent with those of Tungjitkusolmun et al. [7] who reported an increase of 15.3°C in maximum tissue temperature. The discrepancy between these results is explained, both by the fact that Tungjitkusolmun et al. account for conduction of heat into the electrode and that the energy is applied in a power-controlled manner. Furthermore, a round-tipped electrode is used by Tungjitkusolmun et al., while the current study assumes a flat-tipped electrode.

The results of these three numerical models have provided insights into RF ablation and some of the factors influencing lesion geometry. This work has, also,

demonstrated how the analytical solution effectively serves as a benchmark for numerical studies of RF ablation. To better determine the feasibility of using the analytical solution for the inverse problem, it is recommended that additional numerical modeling be done. In particular, it is recommended that a model be developed to account for the resistive heating of the blood, which may significantly affect maximum tissue temperature and lesion distortion. In addition, it is suggested that a single model be developed that incorporates all factors of interest to determine their combined effect on lesion geometry.

REFERENCES

[1] http://rf-ablation.engr.wisc.edu/start_heart.html, 2003

[2] Morady, M., "Radio-frequency ablation as treatment for cardiac arrhythmias," *N. Engl. J. Med.*, vol. 340, no. 7, pp. 534-543, Feb. 1999.

[3] Panescu, D., Whayne, J. G., Fleischman, S. D., Mirotznik, M. S., Swanson, D. K. and Webster, J. G., "Three-dimensional finite element analysis of current density and temperature distributions during radio-frequency ablation," *IEEE Trans. Biomed. Eng.*, vol. 42, no. 9, pp. 879-890, 1995.

[4] Jain, M. K. and Wolf, P. D., "A three-dimensional finite element model of radiofrequency ablation with blood flow and its experimental validation," *Ann. Biomed. Eng.*, vol. 28, pp. 1075-1084, 2000.

[5] Tungjitkusolmun, S., Woo, E. J., Cao, H., Tsai, J., Vorperian, V.R., and Webster, J. G., "Finite element analyses of uniform current density electrodes for radio-frequency cardiac ablation," *IEEE Trans. Biomed. Eng.*, vol. 47, no. 1, pp. 32-39, Jan. 2000.

[6] Labonté, S., "Numerical model for radio-frequency ablation of the endocardium and its experimental validation," *IEEE Trans. Biomed. Eng.*, vol. 41, no. 2, pp.108-115, 1994.

[7] Tungjitkusolmun, S., Woo, E. J., Cao, H., Tsai, J.-Z., Vorperian, V. R., and Webster, J. G., "Thermal-electrical finite element modeling for radio-frequency cardiac ablation: effects of changes in myocardial properties," *Med. Biol. Eng. Comput.*, vol. 38, pp. 562-568, 2000.

[8] Bhavaraju, N. C., *Heat Transfer Modeling During Cardiac Ablation in Swine Myocardium*. *PhD Thesis*, The University of Texas at Austin, Department of Biomedical Engineering, 1999.

[9] Poularikas, A. D., 2000, *The Transforms and Applications Handbook*, CRC Press LLC, Boca Raton.

[10] Kakac, S. and Yener, Y., 1993, *Heat Conduction*, Taylor & Francis, Washington, DC.

[11] Wiley, J. D. and Webster, J. G., "Analysis and control of the current distribution under circular dispersive electrodes," *IEEE Trans. Biomed. Eng.*, vol. 29, no. 5, pp. 381-385, May 1982.

[12] Wheelon, A. D., 1968, *Table of Summable Series and Integrals Involving Bessel Functions*, Holden-Day, San Francisco.

[13] Richards, D., 2002, *Advanced Mathematical Methods with Maple®*, Cambridge University Press, Cambridge.

[14] Press, W. H., 2002, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, Cambridge.

[15] Thakkar, A. J. and Smith, V. H., "A strategy for the numerical evaluation of Fourier sine and cosine transforms to controlled accuracy," *Comp. Phys. Comm.*, vol. 10, pp. 73-79, 1975.

[16] Puoskari, M., "A method for computing Bessel function integrals," *J. Comp. Phys.*, vol. 75, pp. 334-344, 1988.

[17] http://mathworld.wolfram.com/Ellipsoid.html, 2003

[18] Incropera, F. P. and DeWitt, D. P., 2002, *Fundamentals of Heat and Mass Transfer*, John Wiley & Sons, New York.

[19] Özişik, M. N., 1968, *Boundary Value Problems of Heat Conduction*, International Textbook Company, Scranton.

[20] Churchill, R. V., 1958, *Operational Mathematics*, McGraw-Hill Book Company, New York.

APPENDIX A – DERIVATION OF THE APPROPRIATE FORM FOR THE FOURIER TRANSFORM KERNEL

The kernel for the Fourier transform in a semi-infinite region with a convective boundary condition at the surface, shown as Eq. (A.1), may be shown to have the correct form by considering the appropriate auxiliary problem [19]. This auxiliary problem is shown as Eqs. (A.2) and (A.3).

$$K(\beta,\zeta)=\sqrt{\frac{2}{\pi}}\left[\frac{\beta\cos\beta\zeta+Bi\sin\beta\zeta}{\sqrt{\beta^2+Bi^2}}\right] \tag{A.1}$$

$$\frac{d^2Z(\zeta)}{d\zeta^2}+\beta^2Z(\zeta)=0 \qquad 0\le\zeta<\infty \tag{A.2}$$

$$\frac{dZ(\zeta)}{d\zeta}-BiZ(\zeta)=0 \qquad \zeta=0 \tag{A.3}$$

Equation (A.4), below, can be shown to satisfy Eqs. (A.2) and (A.3). The parameter, β, is assumed to take all values from zero to infinity, continuously.

$$Z(\beta,\zeta)=\beta\cos\beta\zeta+Bi\sin\beta\zeta \tag{A.4}$$

Any arbitrary function, $F(\zeta)$, in the semi-infinite interval, $0\le\zeta<\infty$, may be written in terms of the function, $Z(\beta,\zeta)$. Because Eq. (A.4) satisfies the auxiliary problem described by Eqs. (A.2) and (A.3) for all values of β, $F(\zeta)$ is the integral of Eq. (A.4) over the range of β values. This arbitrary function is shown as Eq. (A.5), below.

$$F(\zeta)=\int_0^\infty C(\beta)\left(\beta\cos\beta\zeta+Bi\sin\beta\zeta\right)d\beta \qquad 0\le\zeta<\infty \tag{A.5}$$

The coefficient, $C(\beta)$, has been determined by Churchill [20] by means of the Laplace transformation so that Eq. (A.5) may be written in the form shown in Eq. (A.6), below.

$$F(\zeta)=\frac{2}{\pi}\int_0^\infty\left(\beta\cos\beta\zeta+Bi\sin\beta\zeta\right)\left[\int_0^\infty\frac{\beta\cos\beta\zeta'+Bi\sin\beta\zeta'}{\beta^2+Bi^2}F(\zeta')d\zeta'\right]d\beta \tag{A.6}$$

The form of the kernel shown in Eq. (A.1) is found by comparison of Eq. (A.6) with the expressions for the Fourier integral transform of a function, $F(\zeta)$, in the semi-infinite interval, $0 \le \zeta < \infty$, and its inversion formula. These expressions are shown in Eqs. (A.7) and (A.8), below.

$$F(\zeta) = \int_0^\infty K(\beta, \zeta) \bar{F}(\beta) \, d\beta \qquad (\textit{inversion formula}) \qquad \text{(A.7)}$$

$$\bar{F}(\beta) = \int_0^\infty K(\beta, \zeta') F(\zeta') \, d\zeta' \qquad (\textit{integral transform}) \qquad \text{(A.8)}$$

By comparison of Eq. (A.6) with Eqs. (A.7) and (A.8), it is apparent that the correct form of the kernel, $K(\beta, \zeta)$, for the case of a convective boundary condition at the surface of a semi-infinite medium is that shown above in Eq. (A.1).

APPENDIX B – C CODE USED IN EVALUATING THE ANALYTICAL SOLUTION TO THE BIOHEAT EQUATION

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <stddef.h>
#include <time.h>
#define NR_END 1
#define FREE_ARG char*

#define SINGULARITY_EPS 0.00000001  /***convergence criterion for qrombPrism1 used when
                                       calculating the volume of the surface section containing the
                                       singularity***/
#define EPS 0.00001         /***convergence criterion for qrombPrism1, qrombPrism2, and
                            DoubTranOfHG***/
#define EPS2 0.0005         /***convergence criterion for Temperature***/

#define H 2000              /***convection coefficient***/
#define TAU 5.5             /***dimensionless time***/
#define RGRID 12            /***number of rows in array for load_array2d***/
#define CGRID 12            /***number of columns in array for load_array2d***/
#define RSTART 0.0          /***minimum rho value***/
#define RSTOP 2.4           /***maximum rho value***/
#define CSTART 0.0          /***minimum zeta value***/
#define CSTOP 2.4           /***maximum zeta value***/

#define TEMPERATURE "TemperatureLOG.dat"    /***some output file that used to have a purpose***/
#define LOADARRAY "TemperatureDATA.dat"     /***output file for load_array2d***/

/***files containing forward Hankel and Fourier transform data***/
#define DOUBTRAN1 "CosTranDataMar26.dat"
#define DOUBTRAN2 "SineTranDataMar26.dat"

#define QMAX 2000           /***maximum value for dimensionless heat generation***/

#define RHOSMALL 0.001          /***criterion used by DoubTranOfHG for determining convergence of
                                volume along a row***/
#define ZETASMALL 0.005         /***criterion used by DoubTranOfHG for determining convergence of
                                entire surface volume***/
#define GAMMASMALL 0.0004       /***criterion used by Temperature for determining convergence of
                                volume along a row***/
#define BETASMALL 0.0004        /***criterion used by Temperature for determining convergence of
                                entire surface volume***/

#define PI 3.141592653589793
#define PI2 3.14159
#define SIGMA 0.5           /***tissue electrical conductivity used by HeatGenFUNCTION***/
#define R 0.0013            /***electrode radius used by HeatGenFUNCTION***/
#define VO 28               /***electrode potential used by HeatGenFUNCTION***/
#define KK 0.531            /***tissue thermal conductivity used by HeatGenFUNCTION***/
#define TO 37               /***body temperature used by HeatGenFUNCTION***/
#define BIOT (H*R/KK)       /***Biot number used by Kernel ***/

#define JMAX 12             /***specifies the maximum iteration number for qrombPrism1 and
                            qrombPrism2***/
#define JMAXP (JMAX+1)
```

```
/***values used by DTinterp**********************/
#define TEMPSTEP 5
#define GAMMA_MAX 100
#define BETA_MAX 100
#define MINSTEP (TEMPSTEP/pow(2,JMAX-1))
#define DTINTERP_CONSTANT (20*pow(2,JMAX-1))
/*****************************************/

#define FUNCprism(w, x, y, z) ((*funcPrism)(w, x, y, z))
#define SMALL 0.0000000001      /***used in determining if rho and zeta values are too close to the
                                singularity***/
#define MAXSIZE 2000   /***specifies maximum array size for load_array2d***/

#define K 5         /***number of section volume approximations obtained by qrombPrism1 and
                    qrombPrism2 before using polint to obtain an initial prediction of the actual section
                    volume***/

main(){
        double bessj0(double x);
        void nrerror(char error_text[]);
        double **matrix(long nrl, long nrh, long ncl, long nch);
        void free_matrix(double **m, long nrl, long nrh, long ncl, long nch);
        void load_array2d(double **array2d, double (*function)(double, double, double, double), double
gamma, double beta, double a1, double a2, double b1, double b2, int agrid, int bgrid);
        void print_array2d(double **array2d, int agrid, int bgrid);
        void writeMatrix(double **matrix, FILE* filepointer, double a1, double a2, double b1, double b2,
int agrid, int bgrid);
        void writeMatrix2(double **matrix, FILE* filepointer, double a1, double a2, double b1, double
b2, int agrid, int bgrid);
        double HeatGenFUNCTION(double dummy1, double dummy2,double x, double y);
        double Kernel(double beta, double zeta);
        double *vector(long nl, long nh);
        void free_vector(double *v, long nl, long nh);
        void polint(double xa[], double ya[], int n, double x, double *y, double *dy);
        double prism1(double(*funcPrism)(double, double, double, double), double gamma, double beta,
double LLrho, double ULrho, double LLzeta, double ULzeta, int n);
        double prism2(double(*funcPrism)(double, double, double, double), double gamma, double beta,
double LLrho, double ULrho, double LLzeta, double ULzeta, int n);
        double qrombPrism1(double (*funcQromb)(double, double, double, double), double gamma,
double beta, double LLIMrho, double ULIMrho, double LLIMzeta, double ULIMzeta);
        double qrombPrism2(double (*funcQromb)(double, double, double, double), double gamma,
double beta, double LLIMrho, double ULIMrho, double LLIMzeta, double ULIMzeta);
        double DoubTranOfHG(double dummy1, double dummy2, double gamma, double beta);
        double Temperature(double dummy1, double dummy2, double rho, double zeta);
        double HankFourINTEGRAND(double gamma, double beta, double rho, double zeta);
        double InverseHankFourINTEGRAND(double rho, double zeta, double gamma, double beta);
        double DTinterp(double gamma, double beta);

        int rgrid, cgrid, i;
        double start, stop, difference;
        double rstart, rstop, cstart, cstop;
        double rhodummy=0.0, zetadummy=0.0;
        double **Temperature_data;
        FILE *TemperatureFILE;

        rgrid=RGRID; cgrid=CGRID; rstart=RSTART; cstart=CSTART; rstop=RSTOP; cstop=CSTOP;
```

```
	Temperature_data = matrix(1, rgrid, 1, cgrid);

	/***A file is opened to write the dimensionless time for which temperatures are being
	generated***/
	TemperatureFILE=fopen(TEMPERATURE, "w");
	fprintf(TemperatureFILE, "Generating a temperature profile for tau = %1.2f...\n", TAU);
	fclose(TemperatureFILE);

	start = time(NULL);
	for(i=1;i<=1;i++){
		/***An array, Temperature (a function for evaluating temperatures), and other necessary
		parameters are passed into load_array2d***/
		load_array2d(Temperature_data, Temperature, rhodummy, zetadummy, rstart, rstop,
cstart, cstop, rgrid, cgrid);
	}
	stop = time(NULL);
	difference = difftime(stop,start);

	/***The total time to generate the temperature profile is written to a file***/
	TemperatureFILE=fopen(TEMPERATURE, "a");
	fprintf(TemperatureFILE, "Elapsed time for temperature data: %1.0lf seconds.\n", difference);
	fclose(TemperatureFILE);

	free_matrix(Temperature_data, 1, rgrid, 1, cgrid);
}

double bessj0(double x)
/***Numerical Recipes© function that returns the Bessel function Jo(x) for any real x.***/
{
	double ax,z;
	double xx, y, ans, ans1, ans2;

	if ((ax=fabs(x)) < 8.0){
		y=x*x;
		ans1=57568490574.0+y*(-13362590354.0+y*(651619640.7
		  +y*(-11214424.18+y*(77392.33017+y*(-184.9052456)))));
		ans2=57568490411.0+y*(1029532985.0+y*(9494680.718
		  +y*(59272.64853+y*(267.8532712+y*1.0))));
		ans=ans1/ans2;
	}else{
		z=8.0/ax;
		y=z*z;
		xx=ax-0.785398164;
		ans1=1.0+y*(-0.1098628627e-2+y*(0.2734510407e-4
		  +y*(-0.2073370639e-5+y*0.2093887211e-6)));
		ans2=-0.1562499995e-1+y*(0.1430488765e-3
		  +y*(-0.6911147651e-5+y*(0.7621095161e-6
		  -y*0.934935152e-7)));
		ans=sqrt(0.636619772/ax)*(cos(xx)*ans1-z*sin(xx)*ans2);
	}
	return ans;
}

void nrerror(char error_text[])
/***Numerical Recipes© standard error handler***/
```

```
{
	fprintf(stderr, "Numerical Recipes run-time error...\n");
	fprintf(stderr, "%s\n", error_text);
	fprintf(stderr, "...now exiting to system...\n");
	exit(1);
}

double **matrix(long nrl, long nrh, long ncl, long nch)
/***Numerical Recipes© function that allocates memory for a 2-D array***/
{
	long i, nrow=nrh-nrl+1, ncol=nch-ncl+1;
	double **m;

	m=(double **) malloc((size_t)((nrow+NR_END)*sizeof(double*)));
	if (!m) nrerror("allocation failure 1 in matrix()");
	m += NR_END;
	m -= nrl;

	m[nrl]=(double *) malloc((size_t)((nrow*ncol+NR_END)*sizeof(double)));
	if (!m[nrl]) nrerror("allocation failure 2 in matrix()");
	m[nrl] += NR_END;
	m[nrl] -= ncl;

	for(i=nrl+1;i<=nrh;i++) m[i]=m[i-1]+ncol;

	return m;
}

void free_matrix(double **m, long nrl, long nrh, long ncl, long nch)
/***Numerical Recipes© function that frees memory that has been allocated for a 2-D array using
matrix***/
{
	free((FREE_ARG) (m[nrl]+ncl-NR_END));
	free((FREE_ARG) (m+nrl-NR_END));
}


/***load_array2d accepts inputs specifying the range of rho and zeta values for which dimensionless
temperatures are evaluated as well as integers used to determine the step size between adjacent data points.
These parameters are defined as global constants: RGRID, CGRID, RSTART, RSTOP, CSTART,
CSTOP***/

void load_array2d(double **array2d, double (*function)(double, double, double, double), double gamma,
double beta, double a1, double a2, double b1, double b2, int agrid, int bgrid)
{
	double astep, bstep, x, y;
	int n, m;
	double start, stop;
	static int print=1;
	FILE *LoadArrayFILE;

	if(agrid>MAXSIZE || bgrid>MAXSIZE)
		printf("Grid size must not exceed %i.\n", MAXSIZE);
	else{
		astep = (a2-a1)/agrid;
		bstep = (b2-b1)/bgrid;
```

```
		for(n= 1; n<=agrid; n++){
			x = a1 + (n-1)*astep;

			if(n==1){
				LoadArrayFILE=fopen(LOADARRAY, "a");
				if(print==1) fprintf(LoadArrayFILE, "\t");
				for(m= 1; m<=bgrid; m++){
					if(print==1){
						if(m!=bgrid) fprintf(LoadArrayFILE, "%1.12e\t",
(b1+(m-1)*bstep));
						else fprintf(LoadArrayFILE, "%1.12e\n", (b1+(m-
1)*bstep));
					}
				}
				fclose(LoadArrayFILE);
			}

			for(m= 1; m<=bgrid; m++){

				y = b1 + (m-1)*bstep;

				array2d[n][m] = function(gamma,beta,x,y);

				LoadArrayFILE=fopen(LOADARRAY, "a");
				if(print==1){
					if(m==1) fprintf(LoadArrayFILE, "%1.12e\t", x);

					if(m!=bgrid) fprintf(LoadArrayFILE, "%1.12e\t",
array2d[n][m]);
					else fprintf(LoadArrayFILE, "%1.12e\n", array2d[n][m]);
					}
				fclose(LoadArrayFILE);
				}
			}
		}
	print++;
}


void print_array2d(double **array2d, int agrid, int bgrid)
/***may be used to print an array of any specified size to the screen***/
{
	int j, k;

	for(j= 1; j<=agrid; j++){
		for(k= 1; k<=bgrid; k++){

		if(k==bgrid) printf("%1.3f\n", array2d[j][k]);
			else printf("%1.3f   ", array2d[j][k]);
		}
	}
}

double HeatGenFUNCTION(double dummy1, double dummy2,double x, double y)
/***calculates the dimensionless heat generation for any specified values of rho and zeta***/
```

```
{
        double v1, v2, v1v2, Q;
        double r, z, dVdr, dVdz, q;
        int whichQ;
        float rhigh, rlow, zhigh;

        r = x;
        z = y;

        if(z==0.0 && (r>-1.0 && r<1.0)) whichQ = 0;
        else if(r>(1.0-SMALL) && r<(1.0+SMALL) && z<SMALL) whichQ = 1;
        else whichQ = 2;

        if(r>(1.0-SMALL) && r<(1.0+SMALL) && z<SMALL){
                printf("Oopsie, you're trying to evaluate at rho=1.0 and zeta=0.0\n");
                }

        switch(whichQ){
                case 0:
                        Q = ((4*SIGMA*VO*VO)/(KK*TO*PI*PI))*((1.0/2.0)*(1/(1+r) + 1/(1-r)));

                        if(Q>QMAX) Q = QMAX;

                        return Q;
                        break;

                case 1:

                        Q = QMAX;

                        printf("You are evaluating at rho = 1 and zeta = 0\n");
                        printf("The heat generation is assigned to be %1.0f.\n",Q);

                        return Q;
                        break;

                case 2:
                        v1 = sqrt(z*z + (1+r)*(1+r));
                        v2 = sqrt(z*z + (1-r)*(1-r));
                        v1v2 = (v1+v2)*(v1+v2);

                        dVdr = -(2*(r+1)/v1+2*(r-1)/v2)/(v1v2*sqrt(1-4/v1v2));
                        dVdz = -(2*z/v1 + 2*z/v2)/(v1v2*sqrt(1-4/v1v2));
                        Q = ((4*SIGMA*VO*VO)/(KK*TO*PI*PI))*(dVdr*dVdr + dVdz*dVdz);

                        if(Q>QMAX) Q = QMAX;

                        return Q;
                        break;
        }
}

double Kernel(double beta, double zeta)
/***calculates the kernel for the Fourier transform***/
{
        double Kernel;
```

```
	Kernel=sqrt(2/PI)*((beta*cos(beta*zeta) + BIOT*sin(beta*zeta))/(sqrt(beta*beta+BIOT*BIOT)));

	return Kernel;
}

double *vector(long nl, long nh)
/*** Numerical Recipes© function that allocates a double vector with subscript range v[nl..nh]***/
{
	double *v;

	v=(double *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(double)));
	if (!v) printf("allocation error in vector()");
	return v-nl+NR_END;
}

void free_vector(double *v, long nl, long nh)
/*** Numerical Recipes© function that frees a double vector allocated with vector()***/
{
	free((FREE_ARG) (v+nl-NR_END));
}

void polint(double xa[], double ya[], int n, double x, double *y, double *dy)
/***Given arrays xa[1..n] and ya[1..n], and given a value x, this routine returns a value y, and an error
estimate dy.  If P(x) is a polynomial such that P(xa_i) = ya_i (i = 1, .., n) then the returned value y = P(x)***/
{
	int i, m, ns=1;
	double den, dif, dift, ho, hp, w;
	double *c, *d;

	dif=fabs(x-xa[1]);
	c=vector(1,n);
	d=vector(1,n);
	for(i=1;i<=n;i++){
		if((dift=fabs(x-xa[i]))<dif){
			ns=i;
			dif=dift;
		}
		c[i]=ya[i];
		d[i]=ya[i];
	}
	*y=ya[ns--];
	for(m=1;m<n;m++){
		for(i=1;i<=n-m;i++){
			ho=xa[i]-x;
			hp=xa[i+m]-x;
			w=c[i+1]-d[i];
			if((den=ho-hp)==0.0) nrerror("Error in routine polint");
			den=w/den;
			d[i]=hp*den;
			c[i]=ho*den;
		}
		*y+=(*dy=(2*ns<(n-m) ? c[ns+1] : d[ns--]));
	}
	free_vector(d,1,n);
	free_vector(c,1,n);
```

```
}


/*** prism1 calculates approximate volumes for a surface section for which the limits of integration have
been specified by DoubTranOfHG (integration limits are passed from DoubTranOfHG to qrombPrism1 to
prism1).  It receives an integer from qrombPrism1 that specifies which volume approximation to return.
Note that this function is not called if DTinterp is used instead of DoubTranOfHG ***/

double prism1(double(*funcPrism)(double, double, double, double), double gamma, double beta, double
LLrho, double ULrho, double LLzeta, double ULzeta, int n)
{
	int i, k, iterations;
	double h, side_sum=0.0, center_sum=0.0;
	double rho, zeta, s1, rhostep, zetastep, drho, dzeta, coolbeans;
	static double sum1;

	drho = ULrho-LLrho;
	dzeta = ULzeta-LLzeta;
	rho = LLrho;

	if(n==1){
		rhostep = drho;
		zetastep = dzeta;
	}else{
		rhostep = drho;
		zetastep = dzeta;
		for(k=1;k<n;k++){
			rhostep *= 0.5;
			zetastep *= 0.5;
		}
	}

	if(n==1){
		sum1 = 0.25*(FUNCprism(gamma, beta, LLrho, LLzeta) + FUNCprism(gamma, beta,
LLrho, ULzeta) + FUNCprism(gamma, beta, ULrho, LLzeta) + FUNCprism(gamma, beta, ULrho,
ULzeta));

	}else{
		iterations = 1;
		for(k=1;k<n;k++) iterations *= 2;
		iterations++;

		for(i=1; i<=iterations; i++){
			if((i-1)%2 == 0){
				zeta = LLzeta + zetastep;
				do{
					if((i==1) || (i==iterations)){
						side_sum += FUNCprism(gamma, beta, rho, zeta);
					}else{
						center_sum += FUNCprism(gamma, beta, rho, zeta);
					}
					zeta += 2*zetastep;
				}while(zeta < ULzeta);
			}else{
				zeta = LLzeta;
				do{
```

```
                                        if((zeta < (LLzeta + zetastep/2)) || (zeta>(ULzeta -
zetastep/2))){
                                                side_sum += FUNCprism(gamma, beta, rho, zeta);
                                        }else{
                                                center_sum += FUNCprism(gamma, beta, rho,zeta);
                                        }
                                        zeta += zetastep;
                                }while(zeta<(ULzeta + zetastep/2));
                        }
                rho += rhostep;
                }
        sum1 += (0.5*(side_sum) + center_sum);
        }

        s1 = sum1*rhostep*zetastep;

        return s1;
}


/*** prism2 calculates approximate volumes for a surface section for which the limits of integration have
been specified by Temperature (integration limits are passed from Temperature to qrombPrism2 to
prism2).  It receives an integer from qrombPrism2 that specifies which volume approximation to
return.***/

double prism2(double(*funcPrism)(double, double, double, double), double gamma, double beta, double
LLrho, double ULrho, double LLzeta, double ULzeta, int n)
{
        int i, k, iterations;
        double h, side_sum=0.0, center_sum=0.0;
        double rho, zeta, s2, rhostep, zetastep, drho, dzeta, coolbeans;
        double value1, value2, value3, value4;
        static double sum2;

        drho = ULrho-LLrho;
        dzeta = ULzeta-LLzeta;
        rho = LLrho;

        if(n==1){
                rhostep = drho;
                zetastep = dzeta;
        }else{
                rhostep = drho;
                zetastep = dzeta;
                for(k=1;k<n;k++){
                        rhostep *= 0.5;
                        zetastep *= 0.5;
                }
        }

        if(n==1){
                sum2 = 0.25*(FUNCprism(gamma, beta, LLrho, LLzeta) + FUNCprism(gamma, beta,
LLrho, ULzeta) + FUNCprism(gamma, beta, ULrho, LLzeta) + FUNCprism(gamma, beta, ULrho,
ULzeta));

        }else{
```

```
		iterations = 1;
		for(k=1;k<n;k++) iterations *= 2;
		iterations++;

		for(i=1; i<=iterations; i++){
			if((i-1)%2 == 0){
				zeta = LLzeta + zetastep;
				do{
					if((i==1) || (i==iterations)){
						side_sum += FUNCprism(gamma, beta, rho, zeta);
					}else{
						center_sum += FUNCprism(gamma, beta, rho, zeta);
					}
					zeta += 2*zetastep;
				}while(zeta < ULzeta);
			}else{
				zeta = LLzeta;
				do{
					if((zeta < (LLzeta + zetastep/2)) || (zeta>(ULzeta -
zetastep/2))){
						side_sum += FUNCprism(gamma, beta, rho, zeta);
					}else{
						center_sum += FUNCprism(gamma, beta, rho,zeta);
					}
					zeta += zetastep;
				}while(zeta<(ULzeta + zetastep/2));
			}
		rho += rhostep;
		}
	sum2 += (0.5*(side_sum) + center_sum);
	}

	s2 = sum2*rhostep*zetastep;

	return s2;
}


/*** qrombPrism1 directs and monitors the process of integrating the integrand of the forward Hankel and
Fourier transforms of the dimensionless heat generation between the finite limits of integration provided by
DoubTranOfHG. Using volume approximations obtained from prism1 for the current surface section as
well as the polynomial curve fit routine, polint, qrombPrism1 obtains an appropriately converged section
volume that is returned to DoubTranOfHG. Note that this function is not called if DTinterp is used instead
of DoubTranOfHG ***/

double qrombPrism1(double (*funcQromb)(double, double, double, double), double gamma, double beta,
double LLIMrho, double ULIMrho, double LLIMzeta, double ULIMzeta)
{
	double ss,dss,precision;
	double s[JMAXP], h[JMAXP+1];
	int j,toomany=0;
	static int f1=1;
	FILE *DoubTranLogFILE;

	if((LLIMzeta<SMALL) && (ULIMrho>1.0) && (LLIMrho<1.0))
precision=SINGULARITY_EPS;
```

```
	else precision=EPS1;

	h[1]=1.0;
	for (j=1;j<=JMAX;j++){

		s[j]=prism1(funcQromb,gamma,beta,LLIMrho, ULIMrho, LLIMzeta, ULIMzeta, j);
		if(j>=K){
			toomany++;
			polint(&h[j-K],&s[j-K],K,0.0,&ss,&dss);
			if((fabs(dss)<=precision*fabs(ss)) || (j==JMAX)) return ss;
		}
		h[j+1]=0.25*h[j];
	}
	nrerror("Too many steps in routine qromb. Too bad for you.");
	return 0.0;
}
```

/*** *qrombPrism2* directs and monitors the process of integrating the integrand of the inverse Hankel and Fourier transforms between the finite limits of integration provided by *Temperature*. Using volume approximations obtained from *prism2* for the current surface section as well as the polynomial curve fit routine, *polint*, *qrombPrism2* obtains an appropriately converged section volume that is returned to *Temperature*.***/

```
double qrombPrism2(double (*funcQromb)(double, double, double, double), double gamma, double beta,
double LLIMrho, double ULIMrho, double LLIMzeta, double ULIMzeta)
{
	double ss,dss;
	double s[JMAXP], h[JMAXP+1];
	int j;
	static int f2=1;

	h[1]=1.0;
	for (j=1;j<=JMAX;j++){
		s[j]=prism2(funcQromb,gamma,beta,LLIMrho, ULIMrho, LLIMzeta, ULIMzeta, j);

		if(j>=K){
			polint(&h[j-K],&s[j-K],K,0.0,&ss,&dss);
			if((fabs(dss)<=EPS*fabs(ss)) || (j==JMAX)){

				return ss;
			}
		}
		h[j+1]=0.25*h[j];
	}
	nrerror("Too many steps in routine qromb. Too bad for you.");
	return 0.0;
}
```

/***DoubTranOfHG* evaluates the forward Hankel and Fourier transforms of the dimensionless heat generation function. It does this by adding surface section volumes along rows and then adding these row volumes to obtain a total surface volume. It also determines appropriate section sizes to be integrated by *qrombPrism1*. Note that *DTinterp* may be used instead of *DoubTranOfHG****/

```
double DoubTranOfHG(double dummy1, double dummy2, double gamma, double beta){
```

```
double HankFourINTEGRAND(double gamma, double beta, double rho, double zeta);

int stopr=0, stopz=0, row=1;
double rhosumN=0.0, rhosum=0.0, zetasum=0.0;
double LLrho=0.0, ULrho=0.0, LLzeta=0.0, ULzeta=0.0;
double period, smallstep, bigstep, step;
double distanceUL, distanceLL, ztestvalue;

if(gamma<2 && beta<2) period = PI2;
else{
        if(gamma>=beta) period=2*PI2/gamma;
        if(gamma<beta) period=2*PI2/beta;
}

smallstep=0.2*period;
bigstep=0.8*period;

do{
        if(row==1){
                step=smallstep;
                ULzeta+=step;
        }
        do{
                LLrho=ULrho;
                ULrho+=step;


rhosumN=qrombPrism1(HankFourINTEGRAND,gamma,beta,LLrho,ULrho,LLzeta,ULzeta);
                rhosum+=rhosumN;

                if((fabs(rhosumN)<fabs(EPS*rhosum)) || (fabs(rhosumN)<RHOSMALL)){
                        if(row==1) stopr+=1;
                        else stopr+=2;
                        }
                else stopr=0;

        }while(stopr<=6);

        if(row==1){
                step=bigstep;
                row++;
        }
        ULrho=0.0;
        LLzeta=ULzeta;
        ULzeta+=step;
        zetasum+=rhosum;
        if((fabs(rhosum)<fabs(EPS*zetasum)) || (fabs(rhosum)<ZETASMALL)) stopz++;
        else stopz=0;

        rhosum=0.0;
        stopr=0;

}while(stopz<=3);

stopz=0;
```

```
	return zetasum;
}


/***Temperature evaluates the inverse Hankel and Fourier transforms.  It does this by adding surface
section volumes along rows and then adding these row volumes to obtain a total surface volume. It also
determines appropriate section sizes to be integrated by qrombPrism2***/

double Temperature(double dummy1, double dummy2, double rho, double zeta){
	double InverseHankFourINTEGRAND(double rho, double zeta, double gamma, double beta);

	int stopgamma=1, stopbeta=1, row=1, n=0;
	double gammasumN=0.0, gammasum=0.0, betasum=0.0;
	double LLgamma=0.0, ULgamma=0.0, LLbeta=0.0, ULbeta=0.0;
	double period, step;
	double distanceUL, distanceLL;
	FILE *TemperatureFILE;
	static int x=1;

	if(rho<2 && zeta<2) period = 3;
	else{
		if(rho>=zeta) period=6/rho;
		if(rho<zeta) period=6/zeta;
	}

	step=period;

	do{
		if(n==0) ULbeta+=step;

		do{
			LLgamma=ULgamma;
			ULgamma+=step;


	gammasumN=qrombPrism2(InverseHankFourINTEGRAND,rho,zeta,LLgamma,ULgamma,LLbet
a,ULbeta);
				gammasum+=gammasumN;
				if((fabs(gammasumN)>0.005) &&
(fabs(gammasumN)>fabs(EPS2*gammasum))) stopgamma=1;
				else stopgamma++;

				if(x==1) x++;

		}while(stopgamma<=2);

		if(n==0) n++;
		ULgamma=0.0;
		LLbeta=ULbeta;
		ULbeta+=step;
		betasum+=gammasum;
		if(fabs(gammasum)<1.0e-2 || (fabs(gammasum)<fabs(EPS2*betasum))) stopbeta++;
		else stopbeta=1;
		gammasum=0.0;

		stopgamma=1;
```

```
        }while(stopbeta<=2);

        return betasum;
}

double HankFourINTEGRAND(double gamma, double beta, double rho, double zeta)
/***Evaluates the integrand for the forward Hankel and Fourier transforms of the dimensionless heat
generation.  Note that this function is not called if DTinterp is used instead of DoubTranOfHG***/
{
        double HankFourINT,dummy1, dummy2;
        static int n;

        HankFourINT=rho*bessj0(gamma*rho)*Kernel(beta,zeta)*HeatGenFUNCTION(dummy1,dumm
y2,rho,zeta);

        return HankFourINT;
}

double InverseHankFourINTEGRAND(double rho, double zeta, double gamma, double beta)
/***Evaluates the integrand for the inverse Hankel and Fourier transforms ***/
{
        double InverseHankFourINT,dummy1=0.0,dummy2=0.0;
        double tau;

        if((gamma==0.0) && (beta==0.0)) InverseHankFourINT=0.0;
        else
InverseHankFourINT=gamma*bessj0(gamma*rho)*Kernel(beta,zeta)*DoubTranOfHG(dummy1,dummy2,
gamma,beta)*(1-exp(-(gamma*gamma+beta*beta)*TAU))/(gamma*gamma+beta*beta);

        /*** note that this function may be modified to call DTinterp instead of DoubTranOfHG.  In this
case, previously calculated data for the forward Hankel and Fourier transforms of the dimensionless heat
generation would be read from the files, CosTranDataMar26.dat and SineTranDataMar26.dat ***/

        return InverseHankFourINT;
}


/***DTinterp may be used in place of DoubTranOfHG to read previously calculated values for the forward
Hankel and Fourier transforms of the dimensionless heat generation from the files, CosTranDataMar26.dat
and SineTranDataMar26.dat.  If a needed value is not contained in these files, DTinterp calculates an
approximate value by interpolating between values contained in the files***/

double DTinterp(double gamma, double beta){
        static int call=1, n, ratio;
        static double low, high, step, convert;
        static double **FourTranARRAY;
        double DTinterp, FourCosCoef, FourSinCoef, R1, S1, T1, gammax, betax, denom, num, FourCos,
FourSin;
        int i, j, gammai, betaj, pseudo_gammai, pseudo_betaj, gamma_remainder, beta_remainder;
        FILE *FourCosFILE, *FourSinFILE, *array_check, *DoubTranLogFILE, *InvTranLogFILE;

        if(call==1){
                printf("This is inside the call=1 loop.\n");

                FourCosFILE=fopen(DOUBTRAN1, "r");
```

```
                FourSinFILE=fopen(DOUBTRAN2, "r");

                fscanf(FourCosFILE,"%i",&n);
                printf("call: %i, n: %i\n", call, n);

                FourTranARRAY=matrix(1, n+2, 1, n+2);
                FourTranARRAY[1][1]=0.0;

                fscanf(FourCosFILE,"%lf %lf", &low, &high);
                step=(high-low)/n;

                ratio=DTINTERP_CONSTANT/n;
                convert=DTINTERP_CONSTANT/high;

                for(i=2;i<=n+2;i++){
                        FourTranARRAY[1][i]=(i-2)*step;
                        FourTranARRAY[i][1]=(i-2)*step;
                }

                for(i=2;i<=n+2;i++){
                        for(j=1;j<=n+2;j++){
                                if(j==1){
                                        fscanf(FourCosFILE, "%lf", &FourCos);
                                        fscanf(FourSinFILE, "%lf", &FourSin);
                                }else{
                                        num= sqrt(2/(PI*(FourTranARRAY[1][j] *
FourTranARRAY[1][j] + BIOT*BIOT)));
                                        fscanf(FourCosFILE, "%lf", &FourCos);
                                        fscanf(FourSinFILE, "%lf", &FourSin);
                                        FourTranARRAY[i][j]= num*FourTranARRAY[1][j] *
FourCos + num*BIOT*FourSin;
                                }
                        }
                }

                fclose(FourCosFILE);
                fclose(FourSinFILE);
        }

        pseudo_gammai=(int)(gamma*convert+SMALL);
        pseudo_betaj=(int)(beta*convert+SMALL);
        gamma_remainder=pseudo_gammai%ratio;
        beta_remainder=pseudo_betaj%ratio;
        gammai=(pseudo_gammai-gamma_remainder)/ratio + 2;
        betaj=(pseudo_betaj-beta_remainder)/ratio + 2;

        if((gamma_remainder==0) && (beta_remainder==0)){
                DTinterp = FourTranARRAY[gammai][betaj];
        }else{
                gammax = (double)gamma_remainder/(double)ratio;
                betax = (double)beta_remainder/(double)ratio;
                R1=(FourTranARRAY[gammai+1][betaj]-FourTranARRAY[gammai][betaj])*gammax
+ FourTranARRAY[gammai][betaj];
                S1=(FourTranARRAY[gammai+1][betaj+1]-
FourTranARRAY[gammai][betaj+1])*gammax + FourTranARRAY[gammai][betaj+1];
                DTinterp=(S1-R1)*betax + R1;
```

```
        }

        call++;

        return DTinterp;
}
```

APPENDIX C – USER-DEFINED FUNCTIONS USED FOR NUMERICAL MODELING

UDF USED FOR MODELS 1 AND 2

```
/***UDF used in Models 1 and 2 to specify inlet velocity profiles.  Three different sets of flow conditions
may be specified by changing the values of FREE_STREAM 1, FREE_STREAM 2 and FREE_STREAM
3, for the free-stream blood velocity, and DELTA 1, DELTA 2, and DELTA 3, for the boundary layer
thickness***/

#include "udf.h"

#define SIGMA 0.5
#define QMAX 2.33e10
#define RADIUS 0.0013
#define PI 3.14159265
#define SMALLN 0.00000001
#define HEIGHT 0.02

#define FREE_STREAM_1 0.2
#define DELTA_1 0.02

#define FREE_STREAM_2 0.3
#define DELTA_2 0.01

#define FREE_STREAM_3 0.4
#define DELTA_3 0.005

DEFINE_SOURCE(THREED_energy_source_xz, cell, thread, dS, eqn){
        real source;

        real v1, v2, v1v2, rlow, rhigh;
        real r, z, xyzcoords[ND_ND],dvdr, dvdz, q;
        int whichq;

        C_CENTROID(xyzcoords, cell, thread);
        r=sqrt(pow(xyzcoords[2],2) + pow(xyzcoords[0],2))/RADIUS;
        z=fabs(xyzcoords[1])/RADIUS;

        v1 = sqrt(z*z + (1+r)*(1+r));
        v2 = sqrt(z*z + (1-r)*(1-r));
        v1v2 = (v1+v2)*(v1+v2);

        dvdr = -(2*(r+1)/v1+2*(r-1)/v2)/(v1v2*sqrt(1-4/v1v2));
        dvdz = -(2*z/v1 + 2*z/v2)/(v1v2*sqrt(1-4/v1v2));
        q = (4*SIGMA*28*28/(RADIUS*RADIUS*PI*PI))*(dvdr*dvdr + dvdz*dvdz);
        /*if(q>QMAX) q = QMAX;*/
        dS[eqn]=0.0;

        return q;

}
```

```
DEFINE_PROFILE(inlet_x_velocity_1, thread, nv){
	face_t f;
	real x[ND_ND];

	begin_f_loop(f, thread)
		{
			F_CENTROID(x,f,thread);
			if(x[1]<=DELTA_1){
				F_PROFILE(f,thread,nv) =
FREE_STREAM_1*sin(PI*x[1]/(2*DELTA_1));
			}else{
				F_PROFILE(f,thread,nv) = FREE_STREAM_1;
			}
		}
	end_f_loop(f,thread)

}

DEFINE_PROFILE(inlet_x_velocity_2, thread, nv){
	face_t f;
	real x[ND_ND];

	begin_f_loop(f, thread)
		{
			F_CENTROID(x,f,thread);
			if(x[1]<=DELTA_2){
				F_PROFILE(f,thread,nv) =
FREE_STREAM_2*sin(PI*x[1]/(2*DELTA_2));
			}else{
				F_PROFILE(f,thread,nv) = FREE_STREAM_2;
			}
		}
	end_f_loop(f,thread)

}

DEFINE_PROFILE(inlet_x_velocity_3, thread, nv){
	face_t f;
	real x[ND_ND];


	begin_f_loop(f, thread)
		{
			F_CENTROID(x,f,thread);
			if(x[1]<=DELTA_3){
				F_PROFILE(f,thread,nv) =
FREE_STREAM_3*sin(PI*x[1]/(2*DELTA_3));
			}else{
				F_PROFILE(f,thread,nv) = FREE_STREAM_3;
			}
		}
	end_f_loop(f,thread)

}
```

UDF USED FOR MODEL 3

```
/***UDF used in Model 3 to calculate local heat generation rates.  Given the local temperature, the
electrical conductivity is calculated.  Also, given the local electric potential gradient, the UDF calculates
the heat generation as the product of the electrical conductivity and the square of the electric potential
gradient***/

#include "udf.h"

#define SIGMA 0.5

#define QMAX 2.33e10
#define RADIUS 0.0013
#define PI 3.14159265
#define SMALLN 0.00000001
#define FACTOR 0.02


DEFINE_SOURCE(THREED_energy_source_xy_Tdepen, cell, thread, dS, eqn){
        real q, temperature, elec_cond;
        real grad0, grad1, grad2;
        int i;

        grad0 = NV_MAG(C_UDSI_G(cell,thread,0));

        temperature = C_T(cell, thread);
        elec_cond = SIGMA + FACTOR*SIGMA*(temperature-310);

        q = elec_cond*(grad0*grad0);

        dS[eqn]=0.0;

        return q;

}

DEFINE_DIFFUSIVITY(electrical_conductivity, cell, thread, i)
{
        real temperature;

        temperature = C_T(cell, thread);

        return SIGMA + FACTOR*SIGMA*(temperature-310);
}
```